3/000 1290

#2

日　本　国　特　許　庁
PATENT OFFICE
JAPANESE GOVERNMENT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２０００年　９月１２日

出願番号
Application Number: 特願２０００－２７５９８３

出願人
Applicant (s): 株式会社日立製作所
日立北海セミコンダクタ株式会社

２００１年　４月　６日

特許庁長官
Commissioner,
Patent Office 及川耕造

出証番号　出証特２００１－３０２６８６１

【書類名】　特許願

【整理番号】　H00012901

【提出日】　平成12年 9月12日

【あて先】　特許庁長官殿

【国際特許分類】　G06F 11/10

【発明者】

【住所又は居所】　東京都小平市上水本町五丁目２０番１号　株式会社日立製作所　半導体グループ内

【氏名】　矢田　直樹

【発明者】

【住所又は居所】　北海道亀田郡七飯町字中島１４５番地　日立北海セミコンダクタ株式会社内

【氏名】　石川　栄一

【特許出願人】

【識別番号】　000005108

【氏名又は名称】　株式会社日立製作所

【特許出願人】

【識別番号】　000233594

【氏名又は名称】　日立北海セミコンダクタ株式会社

【代理人】

【識別番号】　100089071

【弁理士】

【氏名又は名称】　玉村　静世

【電話番号】　047-361-8861

【手数料の表示】

【予納台帳番号】　011040

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　データ処理システム及びデータ処理方法

【特許請求の範囲】

【請求項１】　書き換え可能な不揮発性メモリと、中央処理装置とを有し、前記中央処理装置は、所定の処理を実行し、前記不揮発性メモリの指定された一部の記憶領域の書換え保証回数を当該不揮発性メモリのその他の記憶領域の書換え保証回数よりも向上させる処理を行なうものであることを特徴とするデータ処理システム。

【請求項２】　書き換え可能な不揮発性メモリと、中央処理装置とを有し、前記中央処理装置は、所定の処理を実行して、前記不揮発性メモリの指定された一部の記憶領域に対するライトデータに誤り訂正情報を生成して付加し、前記指定された一部の記憶領域からのリードデータに対し誤り訂正情報による誤り判定と誤り訂正が可能であることを特徴とするデータ処理システム。

【請求項３】　１個の半導体チップに前記不揮発性メモリ及び中央処理装置が形成されたシングルチップのマイクロコンピュータであることを特徴とする請求項１又は２記載のデータ処理システム。

【請求項４】　前記不揮発性メモリ及び中央処理装置が夫々別々の半導体チップに形成されたマルチチップ形態であることを特徴とする請求項１又は２記載のデータ処理システム。

【請求項５】　前記所定の処理は、前記不揮発性メモリの指定された一部の記憶領域への書き込みデータに対して誤り訂正情報を生成する誤り訂正情報生成プログラムと、前記指定された一部の記憶領域から読み出された誤り訂正情報付加データに対するエラー判定とエラーの訂正を行なうエラー訂正プログラムであることを特徴とする請求項１又は２記載のデータ処理システム。

【請求項６】　前記データをｎビットとし、ｎビットのデータに対する誤り訂正情報をｍビットとするとき、ｍビットの相互に異なる２進数をｍ＋ｎ列に並べた行列テーブルの記憶領域を有し、前記行列テーブルは前記誤り訂正情報生成プログラム及びエラー訂正プログラムによって参照されるものであることを特徴とする請求項５記載のデータ処理装置。

【請求項７】　前記中央処理装置によってアクセス可能なマスクＲＯＭを有し、前記マスクＲＯＭは、前記誤り訂正情報生成プログラム及びエラー訂正プログラムを保有するものであることを特徴とする請求項５記載のデータ処理システム。

【請求項８】　前記不揮発性メモリのその他の記憶領域は、前記誤り訂正情報生成プログラム及びエラー訂正プログラムの格納領域を有して成るものであることを特徴とする請求項５記載のデータ処理システム。

【請求項９】　前記不揮発性メモリのその他の記憶領域は消去動作が禁止された消去禁止領域と消去及び書き込みが許容された書換え許容領域とを有し、前記前記誤り訂正情報生成プログラム及びエラー訂正プログラムの格納領域は前記消去禁止領域に割当てられて成るものであることを特徴とする請求項８記載のデータ処理システム。

【請求項１０】　前記誤り訂正情報生成プログラムは、誤り訂正情報を生成した後、生成した誤り訂正情報とそれに対応するデータとを規定のフォーマットに従い誤り訂正情報付加データとして前記指定された一部の記憶領域に格納するものであり、前記エラー訂正プログラムは前記規定のフォーマットに従って誤り訂正情報付加データを認識するものであることを特徴とする請求項７又は９記載のデータ処理システム。

【請求項１１】　前記不揮発性メモリのその他の記憶領域は消去動作が禁止された消去禁止領域と消去及び書き込みが許容された書換え許容領域とを有し、前記誤り訂正情報生成プログラム及びエラー訂正プログラムの格納領域は前記書換え許容領域に割り当てられて成るものであることを特徴とする請求項８記載のデータ処理システム。

【請求項１２】　前記不揮発性メモリから前記誤り訂正情報生成プログラム及びエラー訂正プログラムが転送されるＲＡＭを有し、前記中央処理装置は前記ＲＡＭに転送された前記誤り訂正情報生成プログラム及びエラー訂正プログラムを実行するものであることを特徴とする請求項５記載のデータ処理システム。

【請求項１３】　前記中央処理装置はリセットの指示に応答して前記不揮発性メモリから前記ＲＡＭに前記誤り訂正情報生成プログラム及びエラー訂正プロ

グラムを転送するものであることを特徴とする請求項１２記載のデータ処理システム。

【請求項１４】　前記中央処理装置によってアクセス可能なＲＡＭを有し、前記中央処理装置はリセットの指示に応答して、前記不揮発性メモリの一部の記憶領域から順次誤り訂正情報付加データを読み出し、読み出した誤り訂正情報付加データに対し、前記エラー訂正プログラムの実行により前記エラー判定とエラーの訂正を行い、前記エラー判定とエラーの訂正処理を経たデータを前記ＲＡＭに初期的にストアするものであることを特徴請求項５又は７記載のデータ処理システム。

【請求項１５】　前記中央処理装置は、前記エラー訂正プログラムの実行による前記エラー判定処理において訂正不能なエラーの発生を示す情報を外部から認識可能に保持する手段を有して成るものであることを特徴とする請求項５又は７記載のデータ処理システム。

【請求項１６】　演算制御装置のアドレス空間に書換え保証回数の低い第１記憶領域と、書換え保証回数の高い第２記憶領域とを有し、前記第１記憶領域は、前記第２記憶領域への書き込みデータに対してＥＣＣコードを生成するＥＣＣコード生成プログラムと、前記第２記憶領域から読み出されたＥＣＣコード付加データに対するエラー判定とエラーの訂正を行なうエラー訂正プログラムとを有し、前記演算制御装置は、前記第２記憶領域にデータを格納するとき、前記ＥＣＣコード生成プログラムを実行するものであることを特徴とするデータ処理システム。

【請求項１７】　前記演算制御装置は、前記第２記憶領域からデータを読み出すとき、前記エラー訂正プログラムを実行するものであることを特徴とする請求項１６記載のデータ処理システム。

【請求項１８】　前記演算制御装置は、所定の動作モードに応答して前記エラー訂正プログラムを実行して前記第２記憶領域のデータを予めＲＡＭに順次転送可能であることを特徴とする請求項１６記載のデータ処理システム。

【請求項１９】　前記第１記憶領域はマスクＲＯＭであり、前記第２記憶領域は電気的に書き換え可能なフラッシュメモリであることを特徴とする請求項１

６記載のデータ処理システム。

【請求項２０】　前記第１記憶領域及び第２記憶領域は電気的に書換え可能なフラッシュメモリであり、前記フラッシュッメモリは当該フラッシュメモリに対する書き込み消去プログラムを保有し、更に、前記フラッシュメモリから前記書き込み消去プログラムが転送されるＲＡＭを有し、前記演算制御装置は特定の動作モードに応答してＲＡＭ上の前記書き込み消去プログラムを実行可能であることを特徴とする請求項１６記載のデータ処理システム。

【請求項２１】　ｎビットのデータに対してＥＣＣコードをｍビットとするとき、ｍビットの相互に異なる２進数をｍ＋ｎ列に並べた行列テーブルを用いるデータ処理方法であって、ＥＣＣコードの生成では、データの論理値“１”のビット位置に対応する前記行列テーブルの列の値を行方向のビット毎に排他的論理和を採り、これによって得られたｍビットの値をＥＣＣコードとし、データにＥＣＣコードを付加してｍ＋ｎビットの符号語を生成することを特徴とするデータ処理方法。

【請求項２２】　前記符号語の論理値“１”のビット位置に対応する前記行列テーブルの列の値を行方向のビット毎に排他的論理和を採り、これによって得られたｍビットの値が全ビット論理値“０”のときはエラー無と判定して前記符号語のｎビットのデータを正規データとし、前記排他的論理和によって得られたｍビットの値が1ビットでも論理値“１”のときはエラー有りと判定して、前記行列テーブルの列から、前記排他的論理和によって得られたｍビットの２進数に一致する列を検索し、検索された列に対応される位置の符号語のビットを論理値反転して訂正し、訂正された符号語のｎビットのデータを正規データとする、ことを特徴とする請求項２１記載のデータ処理方法。

【請求項２３】　前記中央処理装置は、前記エラー訂正プログラムの実行中にエラー訂正可能なデータを検出した場合、その検出結果に対応する情報を保持する記憶回路を有することを特徴とする請求項５又は７記載のデータ処理システム。

【請求項２４】　上記検出結果に対応する情報は、ワーニング情報として利用されることを特徴とする請求項２３記載のデータ処理システム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、不揮発性メモリに対する書換え保証回数を改善する技術に関し、例えば電気的に書き換え可能なフラッシュメモリを内蔵するマイクロコンピュータに適用して有効な技術に関する。

【０００２】

【従来の技術】

フラッシュメモリなどの電気的に書き換え可能な不揮発性メモリ（以下単にフラッシュメモリとも称する）はメモリセルにプログラムされる閾値電圧の相違に応じて情報を記憶する。閾値電圧の相違はフローティングゲートが保有する電子又は正孔の量の違いによって得られる。斯くフラッシュメモリセルの電子又は正孔の保持性能は書換え回数の増加と共に劣化する。したがって、記憶情報の信頼性という観点から、通常、フラッシュメモリの使用には有限の書換え保証回数が考慮される。

【０００３】

例えば１００回程度の書換え保証回数に対してそれを上回る書換えを可能にするには、１００回の書換え回数毎に、フラッシュメモリの記憶エリアを順次切換え制御して対処することが可能であるが、そのためには、実使用容量に対して数１０～数１０００倍にも及ぶ大きな記憶容量が必要になてしまう。

【０００４】

フラッシュメモリの書換え保証回数を改善するには、デバイス的手法として、ゲート酸化膜を厚くして電子又は正孔の保持性能を上げることが可能である。また、回路的な手法としてＥＣＣ（エラー・チェック・アンド・コレクト）回路を採用することが可能である。特開平１１－２９６３９２号公報には１チップマイクロコンピュータに内蔵のＥＥＰＲＯＭにＥＣＣを適用した技術が示される。

【０００５】

【発明が解決しようとする課題】

しかしながら、前記ゲート酸化膜を厚くすることによる書換え保証回数の改善

には限界が有り、しかも書き込み時間の大幅な増大が予想される。

【０００６】

また、フラッシュメモリにハードウェアでＥＣＣ機能を付加する場合には、ＥＣＣコードの生成回路と、ＥＣＣコード付加データに対する誤り判定及び訂正の回路が設けられる。しかしながら、その場合には、ＥＣＣ用のハードウェアによってチップ面積が増える。その上、必然的に全てのデータに対してＥＣＣコードを付加してデータ記憶を行なうことになるので、それに応じてフラッシュメモリ自体の記憶容量も大きくされることになる。特開平１１－２９６３９２号公報記載の技術はＥＣＣコードの生成をＣＰＵのファームＲＯＭを用いてソフトウェアで行なうことにより、その分の専用ハードウェアは削減されている。一方、ＥＣＣコード付加データに対する誤り判定及び訂正を行なうハードウェア回路は依然設けられている。しかも、全ての記憶データに対してＥＣＣコードを付加してデータ記憶が行なわれるようになっている。

【０００７】

特に本発明者は、組込み機器制御用途のマイクロコンピュータ等におけるフラッシュッメモリの利用形態を考慮した。例えば、フラッシュメモリには、比較的書換え回数の少ないデータとして例えば、回路特性調整用のトリミングデータ、参照用のテーブルデータ、プログラムデータが記憶される。さらにその他に、機器の状態に応じたパラメータデータのような頻繁に書き換えることが必要なデータもフラッシュメモリに記憶されることがある。一つのフラッシュメモリにそれらデータを混在させようとするとき、従来の技術では全てのデータに対してＥＣＣコードが付加されるため、記憶エリアの利用効率が著しく悪化してしまう。頻繁に書き換えるべきデータ量が少ない場合には特に記憶エリアの利用効率の低下が顕著となるであろう。不揮発性メモリの記憶容量が増大する傾向にあるとき、本発明者は一部の記憶領域だけをＥＣＣの対象にするという観点の有用性を見出した。

【０００８】

また、フラッシュメモリのメモリセル特性はプロセスばらつきの影響を受けるので、同じ書換え回数でも読み出しエラーを生じ易いメモリセルと生じ難いメモ

リセルがある。本発明者は、そのメモリセル特性の個体差に着目して、データビット数に対するＥＣＣコードのビット数をフラッシュメモリの特性に応じて決定できるようにすることの有用性を見出した。要するに、一定の書換え保証回数を得るのに、メモリセルのデバイス特性に合ったＥＣＣ方式を選択できるようにして、データに対するＥＣＣコードのオーバヘッドを小さくして記憶領域の利用効率を最大限とすることの有用性を見出した。本発明者の検討によれば、このＥＣＣ方式の選択という融通性は、ＥＣＣコードの生成又は誤り検出及び訂正をハードウェアで行なう場合には容易に実現し難いということが明らかにされた。

【０００９】

本発明の目的は、誤り訂正情報例えばＥＣＣコードによる記憶領域の利用の無駄を省いて記憶情報の信頼性を向上させることが可能なデータ処理システムを提供することにある。

【００１０】

本発明の別の目的は、ＥＣＣコードによる記憶領域の利用の無駄を省いて記憶情報の書換え保証回数を向上させることが可能なデータ処理システムを提供することにある。

【００１１】

本発明の更に別の目的は、デバイスの特性に合ったＥＣＣ方式を選択できるようにして、データに対するＥＣＣコードのオーバヘッドを小さくして記憶領域の利用効率を最大限とすることが可能なデータ処理システムを提供することにある。

【００１２】

本発明の更に別の目的は、ＥＣＣコードが付加されたデータに対する誤り判定及び訂正の処理によるデータリード動作の遅延を最小限にすることが可能なデータ処理システムを提供することにある。

【００１３】

本発明のその他の目的は、ＥＣＣコードの生成を効率的に行なうことができるデータ処理方法を提供することにある。

【００１４】

本発明のその他の目的は、ＥＣＣコードが付加されたデータの誤り判定を効率的に行なうことができるデータ処理方法を提供することにある。

【００１５】

本発明のその他の目的は、電気的に消去及びプログラム可能な不揮発性メモリのアドレス空間の一部のアドレス領域の書き換え回数をソフトウェア的処理によって向上することが可能なデータ処理システムないしデータ処理方法を提供することにある。

【００１６】

本発明の前記並びにその他の目的と新規な特徴は本明細書の記述及び添付図面から明らかになるであろう。

【００１７】

【課題を解決するための手段】

本願において開示される発明のうち代表的なものの概要を簡単に説明すれば下記の通りである。

【００１８】

〔１〕データ処理システムは、書き換え可能な不揮発性メモリ（フラッシュメモリ）と、中央処理装置（ＣＰＵ）とを有し、前記中央処理装置は、所定の処理（プログラム）を実行し、前記不揮発性メモリのアドレス空間内の指定された一部の記憶領域（２０Ｂａ）の書換え保証回数を当該不揮発性メモリのアドレス空間内のその他の記憶領域の書換え保証回数よりも向上させる処理が可能である。前記所定のプログラムの実行による処理は、前記指定された一部の記憶領域に対するライトデータに誤り訂正情報（ＥＣＣコード）を生成して付加する処理であり、前記不揮発性メモリの指定された一部の記憶領域からのリードデータに対しＥＣＣコードによる誤り判定と誤り訂正を行なう処理である。前記一部の記憶領域の指定は例えばユーザプログラムによって行なわれる。

【００１９】

上記より、不揮発性メモリの指定された一部の記憶領域に格納されたデータに対するアクセスだけを対象として、ＥＣＣコードの付加や誤り訂正を行なって書換え保証回数を向上させる。前記指定された一部の記憶領域には書換えの頻繁な

パラメータデータが格納され、他の記憶領域には書換え頻度の低いプログラムデータなどが格納される。したがって、当該他の記憶領域に格納されるデータにはＥＣＣコードが付加されない。この構成は、記憶領域に拘わらず全てのライトデータに対して区別なくＥＣＣコードを付加する構成に比べて、実質的に無用のＥＣＣコードによる記憶領域の無駄な利用を省きながら記憶情報の信頼性を向上、若しくは記憶情報の書換え保証回数の向上を実現する。

【００２０】

ライトデータに対するＥＣＣコードの付加、並びにＥＣＣコードによる誤り判定及び訂正の処理を中央処理装置によるプログラム実行で実現するから、データビット数に対するＥＣＣコードのビット数の割合を定義するＥＣＣ方式はプログラムの記述内容で選択できる。そのため、メモリセルのデバイス特性に合ったＥＣＣ方式の選択が容易であり、データに対するＥＣＣコードのオーバヘッドを小さくできる。その結果記憶領域の利用効率を最大限とすることが可能になる。要するに、ソフトウェアによりエラー訂正効率を容易に変えることができるから、デバイスの能力に合ったエラー訂正方式を容易に選択することができる。観点を変えれば、これはＥＣＣコードのビット数から無駄を排除し、記憶エリアの有効利用を保証する。

【００２１】

前記データ処理システムは、１個の半導体チップに前記不揮発性メモリ及び中央処理装置が形成されたシングルチップのマイクロコンピュータとして実現することが可能である。一方、前記データ処理システムは、前記不揮発性メモリ及び中央処理装置が夫々別々の半導体チップに形成されたマルチチップ形態で実現してもよい。

【００２２】

中央処理装置が実行する前記所定のプログラムは、例えば、前記不揮発性メモリの指定された一部の記憶領域への書き込みデータに対してＥＣＣコードを生成するＥＣＣコード生成プログラムと、前記不揮発性メモリの指定された一部の記憶領域から読み出されたＥＣＣコード付加データに対するエラー判定とエラーの訂正を行なうエラー訂正プログラムである。

【００２３】

ＥＣＣによる符号化や誤り訂正の公知の手法では例えばハミングコードの検査行列を用い、誤り訂正ではリードデータとハミングコードの行列演算を積和演算などを用いて行なうことができる。ここでは、データをｎビットとし、ｎビットのデータに対するＥＣＣコードをｍビットとするとき、ｍビットの相互に異なる２進数をｍ＋ｎ列に並べた行列テーブルを記憶領域に形成し、前記ＥＣＣコード生成プログラム及びエラー訂正プログラムの処理にその行列テーブルを参照させる。

【００２４】

例えば、ＥＣＣコードの生成では、データの論理値“１”のビット位置に対応する前記行列テーブルの列の値を行方向のビット毎に排他的論理和を採り、これによって得られたｍビットの値をＥＣＣコードとし、データにＥＣＣコードを付加してｍ＋ｎビットの符号語を生成する。エラー判定及びエラー訂正では、前記符号語の論理値“１”のビット位置に対応する前記行列テーブルの列の値を行方向のビット毎に排他的論理和を採り、これによって得られたｍビットの値が全ビット論理値“０”のときはエラー無と判定して前記符号語のｎビットのデータを正規データとし、前記排他的論理和によって得られたｍビットの値が１ビットでも論理値“１”のときはエラー有りと判定して、前記行列テーブルの列から、前記排他的論理和によって得られたｍビットの２進数に一致する列を検索し、検索された列に対応される位置の符号語のビットを論理値反転して訂正し、訂正された符号語のｎビットのデータを正規データとする。このエラー判定では積和演算と除算演算のような処理を要せず、積和演算器及び除算器を備えずともソフトウェアによるＥＣＣ処理を能率的に行なうことが可能になる。

【００２５】

前記前記ＥＣＣコード生成プログラム及びエラー訂正プログラムは前記中央処理装置によってアクセス可能なマスクＲＯＭに保持させてよい。それらプログラムの保持に前記不揮発性メモリを利用する場合には、その他の記憶領域に保持させればよい。パラメータに比べてプログラムの書換え頻度は少ないから相対的に書換え保証回数の小さなその他の記憶領域に保持させれば充分である。要するに

、劣悪な使用条件を除けば、書換え頻度の低い情報にＥＣＣコードを付加する実益は殆ど無いということである。

【００２６】

前記不揮発性メモリのその他の記憶領域に、消去動作が禁止された消去禁止領域（２０Ａ）と消去及び書き込みが許容された書換え許容領域（２０Ｂ）とを割当てるとき、消去禁止領域に前記ＥＣＣコード生成プログラム及びエラー訂正プログラムを格納してよい。その場合には、一旦書き込まれたプログラムの書換えは原則不可能であるから、望ましくは、半導体集積回路化されたマイクロコンピュータのようなデータ処理装置の製造メーカが書き込むのがよい。そうであるなら、データとＥＣＣコードとのフォーマットは予め固定的に決定された方が望ましく、前記ＥＣＣコード生成プログラムは、ＥＣＣコードを生成した後、生成したＥＣＣコードとそれに対応するデータとを規定のフォーマットに従いＥＣＣコード付加データとして前記不揮発性メモリの一部の記憶領域に格納し、エラー訂正プログラムは前記規定のフォーマットに従ってＥＣＣコード付加データを認識すればよい。

【００２７】

また、前記ＥＣＣコード生成プログラム及びエラー訂正プログラムを前記書換え許容領域に保持させてもよい。この場合は、ユーザによるそれらプログラムの書き込みを想定する。したがって、データとＥＣＣコードとのフォーマットについてはユーザが作成したプログラムに依存した任意フォーマットであってもよい。製造メーカーがそれらプログラムを提供しようとする場合にもユーザの使い勝手を考慮すれば、Ｃ言語のような高級言語で記載されたソースプログラムで与えるのがよい。データとＥＣＣコードとを別々の配列として把握することも可能になる。

【００２８】

ＥＣＣ処理速度を向上させる一つの手段として、ランダムアクセスメモリ（ＲＡＭ）のアクセスサイクルが前記不揮発性メモリのアクセスサイクルより速い場合、前記不揮発性メモリから前記ＥＣＣコード生成プログラム及びエラー訂正プログラムをＲＡＭ転送し、前記中央処理装置には前記ＲＡＭに転送された前記Ｅ

ＣＣコード生成プログラム及びエラー訂正プログラムを実行させればよい。このとき、前記中央処理装置はリセットの指示に応答して前記不揮発性メモリから前記ＲＡＭに前記ＥＣＣコード生成プログラム及びエラー訂正プログラムを転送するとよい。

【００２９】

ＥＣＣ処理速度を向上させる第２の手段として、前記中央処理装置によってアクセス可能なＲＡＭを有し、前記中央処理装置はリセットの指示に応答して、前記不揮発性メモリの一部の記憶領域から順次ＥＣＣコード付加データを読み出し、読み出したＥＣＣコード付加データに対し、前記エラー訂正プログラムの実行により前記エラー判定とエラーの訂正を行い、前記エラー判定とエラーの訂正処理を経たデータを前記ＲＡＭに初期的にストアするとよい。その後、ＣＰＵはＲＡＭから必要なデータをリードすればよく、リード動作に際してその都度エラー判定を行なうことを要しない。尚、ＣＰＵによる上記エラー訂正プログラムの実行する方法としては、上記不揮発性メモリから直接上記エラー訂正プログラムを読み込んで実行する方法又はＲＡＭに転送された上記エラー訂正プログラムを読み込んで実行する方法のいずれかを採用することが可能である。

【００３０】

前記エラー訂正プログラムの実行による前記エラー判定処理において訂正不能なエラーが発生したき、誤動作防止の観点より、前記中央処理装置は、前記エラー訂正プログラムの実行による前記エラー判定処理において訂正不能なエラーの発生を示す情報を外部から認識可能にレジスタ（ＲＥＲ）、或いはメモリのフラグ領域（３０）に保持させるとよい。ユーザプログラムを介してその領域を所定インターバル毎に参照すれば、訂正不能なエラー発生を認識でき、例えば、訂正不能なエラーを発生したデータブロックの書換え等を実行して対処することが可能になる。

【００３１】

また、前記中央処理装置は前記エラー訂正プログラムの実行によりエラー訂正可能なデータを検出した場合にも、それを示す情報を外部から認識可能に汎用レジスタやメモリのフラグ領域に保持させてよい。そのような情報は上記と同じく

ワーニング情報として利用すればよい。これにより、ユーザシステムは、データが壊れかけている、ということを逸早く認識することができ、壊れかけているデータの書換え（再書込み）を促してデータの信頼性を更に向上させることが可能になる。

【００３２】

〔２〕本発明の別の観点によるデータ処理システムは、演算制御装置のアドレス空間に書換え保証回数の低い第１記憶領域（２０Ｂｂ）と、書換え保証回数の高い第２記憶領域（２０Ｂａ）とを有し、前記第１記憶領域は、前記第２記憶領域への書き込みデータに対してＥＣＣコードを生成するＥＣＣコード生成プログラムと、前記第２記憶領域から読み出されたＥＣＣコード付加データに対するエラー判定とエラーの訂正を行なうエラー訂正プログラムとを有し、前記演算制御装置は、前記第２記憶領域にデータを格納するとき、前記ＥＣＣコード生成プログラムを実行する。

【００３３】

この観点では、第２記憶領域に対する書き込み動作に対してだけＥＣＣコードをソフトウェアで生成するから、上記同様に、実質的に無用のＥＣＣコードによる記憶領域の無駄な利用を省きながら記憶情報の信頼性を向上、若しくは記憶情報の書換え保証回数の向上を実現することができる。更に、ライトデータに対するＥＣＣコードの付加をプログラム実行で実現するから、やはり、不揮発性メモリのメモリセルのデバイスの特性に合ったＥＣＣ方式の選択が容易であり、データに対するＥＣＣコードのオーバヘッドが小さくでき、記憶領域の利用効率を最大限とすることが可能にある。

【００３４】

前記演算制御装置は、前記第２記憶領域からデータを読み出すとき、前記エラー訂正プログラムを実行する。

【００３５】

前記演算制御装置は、所定の動作モードに応答して前記エラー訂正プログラムを実行して前記第２記憶領域のデータを予めＲＡＭに順次転送可能である。

【００３６】

前記第１記憶領域及び第２記憶領域は例えば電気的に書換え可能なフラッシュメモリである。ＥＣＣコード生成プログラム及びエラー訂正プログラムの書換えを考慮すると、書き込み消去プログラムを前記フラッシュメモリに予め格納し、所定の動作モードに応答してこれをＲＡＭに内部転送し、ＲＡＭ上の書き込み消去プログラムを実行してＥＣＣコード生成プログラム及びエラー訂正プログラムの書換えを行なうようにしてよい。或いは、所定の動作モードに応答して外部からＲＡＭに書き込み消去プログラムを転送し、ＲＡＭ上の書き込み消去プログラムを実行してＥＣＣコード生成プログラム及びエラー訂正プログラムの書換えを行なうようにしてよい。

【００３７】

〔３〕ＥＣＣコードを用いた符号語を生成するデータ処理方法では、ｎビットのデータに対してＥＣＣコードをｍビットとするとき、ｍビットの相互に異なる２進数をｍ＋ｎ列に並べた行列テーブルを利用し、ＥＣＣコードの生成では、データの論理値“１”のビット位置に対応する前記行列テーブルの列の値を行方向のビット毎に排他的論理和を採り、これによって得られたｍビットの値をＥＣＣコードとし、データにＥＣＣコードを付加してｍ＋ｎビットの符号語を生成する。

【００３８】

上記符号語に対するエラー判定を行なうデータ処理方法では、前記符号語の論理値“１”のビット位置に対応する前記行列テーブルの列の値を行方向のビット毎に排他的論理和を採り、これによって得られたｍビットの値が全ビット論理値“０”のときはエラー無と判定して前記符号語のｎビットのデータを正規データとし、前記排他的論理和によって得られたｍビットの値が1ビットでも論理値“１”のときはエラー有りと判定して、前記行列テーブルの列から、前記排他的論理和によって得られたｍビットの２進数に一致する列を検索し、検索された列に対応される位置の符号語のビットを論理値反転して訂正し、訂正された符号語のｎビットのデータを正規データとする。このエラー判定では積和演算ないし除算演算のような処理を要せず、算術論理演算器やシフタなどの演算器及び除算器を有し積和演算器を備えていないＣＰＵを用いてもソフトウェアによるＥＣＣ処理

を能率的に行なうことが可能になる。

【００３９】

【発明の実施の形態】

《マイクロコンピュータ》

図１には本発明の一例に係るシングルチップのマイクロコンピュータが示される。同図に示されるマイクロコンピュータ１は、特に制限されないが、単結晶シリコンのような１個の半導体基板（半導体チップ）にＣＭＯＳ集積回路製造技術により形成される。

【００４０】

マイクロコンピュータ１は、演算制御装置としての中央処理装置（ＣＰＵ）２、ＲＡＭ３、バスステートコントローラ（ＢＳＣ）４、電気的に書き換え可能な不揮発性メモリとしてのフラッシュメモリ５、フラッシュコントロールモジュール６、及びその他の内蔵回路を総称するその他モジュール７を有する。その他モジュール７としてマスクＲＯＭ８、割り込みコントローラ（ＩＮＴＣ）９、タイマ（ＴＭＲ）１０、入出力ポート（Ｉ／Ｏ）１１及びシリアルインタフェースコントローラ（ＳＣＩ）１２等を有する。それら回路モジュールはバスＩＡＢ，ＩＤＢ，ＰＡＢ，ＰＤＢ，ＣＯＮＴを介してインタフェースされる。

【００４１】

前記バスＩＡＢ，ＩＤＢは情報伝送速度の比較的速い内部アドレスバス、内部データバスである。前記バスＰＡＢ，ＰＤＢは情報伝送速度が比較的遅い周辺アドレスバス、周辺データバスである。バスＣＯＮＴはバスアクセス制御信号やタイミング制御信号等を伝達する制御信号線を総称する。内部バスＩＤＢ，ＩＡＢと周辺バスＰＤＢ，ＰＡＢとの動作速度の相違若しくはアクセス対象に固有のアクセス形態の相違に対して前記ＢＳＣ４がアクセス動作タイミング等を最適制御すると共に、前記ＢＳＣ４はアクセスアドレスに応じたチップ選択若しくはモジュール選択制御等も行なう。

【００４２】

前記ＣＰＵ２は、特に制限されないが、マスクＲＯＭ８又はＲＡＭ３から命令をフェッチし、フェッチした命令を解読して実行する。ＲＡＭ３はＣＰＵ２のワ

ーク領域若しくはデータ又はプログラムの一時記憶領域とされる。前記マスクＲＯＭ８はプログラム又はデータテーブルなどの記憶領域とされる。割り込みコントローラ１０はマイクロコンピュータ１の外部から与えられる割込要求又はマイクロコンピュータ１内部の状態に応じて内蔵回路モジュールから発生される割込要求を受け、割り込み優先度及び割り込みマスク等に従って割込要求の受付を調停する。割込要求が受付けられると、ＣＰＵ２に割込み信号ＩＲＱが与えられ、割り込みベクタによって割り込み要因がＣＰＵ２に与えられる。ＣＰＵ２は割り込みベクタによって指示されるプログラムに処理を分岐する。Ｉ／Ｏ１１は外部アドレスバス及び外部データバスへの接続、ＳＣＩ１２の外部インタフェース、ＴＭＲ１０の外部イベント信号入力等に用いられる。

【００４３】

前記ＣＰＵ２の具体例は図２に示される。ＣＰＵ２は、特に制限されないが、シフタＳＦＴ及び算術論理演算器ＡＬＵ等の演算器と、３２ビットの汎用レジスタＲ０～Ｒ３１、プログラムカウンタＰＣ、コンディションコードレジスタＣＣＲ及びテンポラリレジスタＴＲ等のレジスタ群、そしてリードデータバッファＲＤＢ、ライトデータバッファＷＤＢ及びアドレスバッファＡＢなどのバッファ回路を実行部に有する。命令制御部は命令レジスタＩＲ、命令デコーダＩＤＥＣ、命令シーケンスロジックＩＮＴＬを有する。

【００４４】

前記プログラムカウンタＰＣは次に実行すべき命令アドレスを保有し、その命令アドレスがアドレスバッファＡＢから内部アドレスバスＩＡＢに出力されると、ＲＡＭ３等の対応アドレスからリードされた命令が内部データバスＩＤＢを介して命令レジスタＩＲにフェッチされる。命令デコーダＩＤＥＣは命令レジスタＩＲの命令を解読して、ＣＰＵ２内部の制御信号を生成して、前記実行部による演算処理を制御する。命令シーケンスロジックＩＮＴＬは割込み信号ＩＲＱ等に応答して命令実行順序を変更する制御を行なう。

【００４５】

図１においてフラッシュメモリ５は、メモリセルアレイ２０、Ｘデコーダ・ドライバ（ＸＤＥ・ＤＶ）２１、センスアンプアレイ（ＳＡＡ）２２、Ｙスイッチ

アレイ（ＹＳＷ）２３、Ｙデコーダ（ＹＤＥ）２４、入出力回路（ＩＦＢ）２５、電源回路（ＶＧＮ）２６、及びタイミングジェネレータ（ＴＧＮ）２７を有する。メモリセルアレイ２０はマトリクス配置されたフラッシュッメモリセル（図示せず）を有する。フラッシュメモリセルは、特に制限されないが、半導体基板若しくはウェル領域にソース、ドレインを有し、チャネルの上方に夫々絶縁膜を介してフローティングゲート及びコントロールゲートが形成されたスタック構造を有し、ソースをソース線に、ドレインをビット線に、コントロールゲートをワード線に接続して構成される。

【００４６】

フラッシュメモリセルは閾値電圧がプログラム可能にされ、プログラムされた閾値電圧に応じて情報を保持する。例えば、１個のフラッシュメモリセルが１ビットの情報を保持する場合に、相対的に高い閾値電圧状態を書き込み状態、相対的に低い閾値電圧状態を消去状態と称する。書き込み状態を得る為の書き込み動作は、特に制限されないが、コントロールゲートに１０Ｖ、ドレインに例えば５Ｖ、ソースおよび基板に例えば０Ｖを印加して、ドレイン・ソース間に電流を流し、これによってホットエレクトロン注入が起こり、フローティングゲートに電子が蓄積され、メモリセルの閾値電圧が高くなる。前記消去状態を得る為の消去動作は、特に制限されないが、コントロールゲートに１０Ｖ、ソース及び基板に例えば－１０Ｖを印加し、さらにドレインを例えば開放（フローティング）にして、フローティングゲートに蓄積された電子を基板に放出させ、これによってメモリセルの閾値電圧が低くなる。

【００４７】

前記入出力回路２５はバスＩＡＢ，ＩＤＢ，ＰＡＢ，ＰＤＢ，ＣＯＮＴとの間でアドレス、制御信号及びコマンドを入力すると共にデータの入出力を行なう。入出力回路２５に入力されたアドレス信号はＸＤＥＣ・ＤＶ２１及びＹＤＥ２４に入力されて夫々デコードされる。ＸＤＥＣ・ＤＶ２１はそのデコード結果に従ってワード線を選択する。ＹＤＥ２４はそのデコード結果に従ってＹＳＷ２３を介してビット線を選択する。ワード線選択及びビット線選択によってフラッシュメモリセルが選択される。読み出し動作では、前記選択されたフラッシュメモリ

セルの読み出しデータは、ＳＡＡ２２にて検出され、入出力回路２５を経てバスＰＤＢまたはＩＤＢに出力される。書き込み動作では、バスＰＤＢ又はＩＤＢから入出力回路２５に与えられる書き込みデータが入出力回路２５内の書き込みラッチ回路にラッチされ、ワード線選択されたメモリセルに対し、ラッチデータに従って書き込み・書き込み阻止が制御される。書き込み処理の前には予めブロック単位でフラッシュメモリセルに対する消去が行なわれる。

【００４８】

前記電源回路２６はクランプ回路やチャージポンプ回路などを有し、フラッシュメモリの書き込み・消去・読み出しなどの動作で使用する様々な電圧を供給する。前記タイミングジェネレータ２７は、制御バスＣＯＮＴを介して供給されるストローブ信号及びデータバスＰＤＢ，ＩＤＢを介して入力されるコマンドに基づいてフラッシュメモリの内部タイミング信号を生成する。

【００４９】

図１において前記フラッシュコントロールモジュール６は、フラッシュメモリ５に対する書き込み及び消去のためのシーケンス制御並びにＥＣＣ処理に利用される回路ブロックである。このフラッシュコントロールモジュール６は、ＣＰＵ２によってアクセス可能な夫々３２ビットの、書き込み／消去制御レジスタＦＬＭＣＲ、消去ブロック指定レジスタＥＢＲ、データレジスタＦＭＰＤＲ０，ＦＭＰＡＲ０、及びリザルトレジスタＦＰＦＲ等の制御レジスタを備え、更に、フラッシュメモリに対する書き込み及び消去のシーケンス動作を制御するシーケンス制御回路２９を有する。

【００５０】

前記書き込み／消去制御レジスタＦＬＭＣＲはフラッシュメモリ５の動作モードを制御するレジスタであり、書き込みの有効／無効を指示するライトイネーブルビットＷＥ、消去動作を指示するイレーズビットＥ、消去ベリファイ動作を指示するイレーズベリファイビットＥＶ、書き込み動作を指示するプログラムビットＰ、書き込みベリファイ動作を指示するプログラムベリファイビットＰＶ、フラッシュメモリの書き込み動作中にエラーの発生したことを示す書き込みエラービットＰＥＲ、消去動作中にエラーの発生したことを示す消去エラービットＥＥ

Ｒ、フラッシュメモリのリード動作時にエラーの発生したことを示すリードエラービットＲＥＲ等を有する。消去ブロック指定レジスタＥＢＲはフラッシュッメモリセルアレイ２０の消去エリアをブロック毎に設定するレジスタであり、ブロック毎にイレーズブロックビットＥＢ０～ＥＢ９を有する。データレジスタＦＭＰＤＲ及びリザルトレジスタＦＰＦＲ等は後述のＥＣＣ処理で利用されるレジスタである。尚、レジスタＦＭＰＤＲ及びＦＰＦＲは、汎用レジスタ（Ｒ０－Ｒ３１）内のレジスタを利用しても良い。

【００５１】

《一部のエリアに対するソフトウェアＥＣＣ処理》

次にマイクロコンピュータ１のＥＣＣ機能について説明する。フラッシュメモリ５のメモリセルアレイ２０は、特に制限されないが、図３の（Ａ）に例示されるようにブート領域２０Ａとユーザ領域２０Ｂに大別される。特に制限されないが、ブート領域２０Ａはマイクロコンピュータ１のユーザによる自由な書換えが禁止される領域であり、ユーザ領域２０Ｂはユーザによる自由な書換えが許容される領域である。要するに、前記消去ブロック指定レジスタＥＢＲのイレーズブロックビットＥＢ０～ＥＢ９の設定によって消去可能なエリアはユーザ領域２０Ｂに限られる。ブート領域に格納されたプログラムはユーザが所定の動作モードを設定することにより実行可能である。尚、フラッシュメモリに対する書換えシーケンスの具体例についてここでは詳述しないが、書き込みデータにしたがった書き込み動作は先に消去動作が完了されることを条件とするものであり、消去ブロックによる消去が指示されなければ当然書き込みデータに従った書き込み動作も行なわれないようになっている。そのような書き込み・消去制御は前記シーケンス制御回路２９が行なう。

【００５２】

マイクロコンピュータ１のＥＣＣ機能は、図３の（Ｂ）に例示されるように、前記ユーザ領域２０Ｂに指定された一部の領域２０Ｂａのデータに向けられている。即ち、ＥＣＣ機能は、ＣＰＵ２が、所定のプログラムを実行することにより、ユーザ領域２０Ｂの一部の記憶領域（第１領域）２０Ｂａの書換え保証回数を当該ユーザ領域２０Ｂのその他の記憶領域（第２領域）２０Ｂｂの書換え保証回

数よりも向上させる機能である。この所定のプログラム実行によるＥＣＣ機能は、前記指定された一部の記憶領域２０Ｂａに対するライトデータにＥＣＣコードを生成して付加する処理（ＥＣＣコード生成処理）と、前記指定された一部の記憶領域２０Ｂａからのリードデータに対しＥＣＣコードによるエラー判定とエラー訂正を行なう処理（ＥＣＣエラー判定訂正処理）とによって実現される。前者の処理はＣＰＵ２がＥＣＣ生成プログラム２１を実行することにより行なわれ、後者の処理はＣＰＵ２がエラー訂正プログラム２２を実行することにより行なわれる。

【００５３】

前記領域２０Ｂａには頻繁に書き換えられる制御用パラメータなどのデータの記憶に利用される。頻繁に書き換えられる制御用パラメータとは、たとえばマイクロコンピュータ１が自動車のエンジン制御に用いられる場合、エンジン停止時の各ピストンの相対位置、エンジン停止直前までの燃費情報等の情報とされる。頻繁に書換えられるデータにＥＣＣ処理を施せば、仮に、書換え回数の増加によるフラッシュメモリセルの特性劣化に起因して読み出しデータに誤りを生じても、ＥＣＣコードによる誤り訂正能力の範囲で、その誤りを訂正することができる。換言すれば、ＥＣＣ処理により実質的に書き換え保証回数を改善若しくは向上させることができる。

【００５４】

前記領域２０Ｂｂは、図３の（Ｂ）では、頻繁に書き換えられることのない情報として、テーブルデータなどの固定データ、前記ＥＣＣ生成プログラム２１、エラー訂正プログラム２２及び他のユーザプログラム等の記憶領域として利用される。前記ＥＣＣ生成プログラム２１及びエラー訂正プログラム２２は、図３の（Ｃ）に例示されるようにブート領域２０Ａに格納し、或いは図３の（Ｄ）に例示されるようにマスクＲＯＭ８に格納してもよい。

【００５５】

上記より、ユーザ領域２０Ｂの一部の記憶領域２０Ｂａに対するアクセスだけを対象として、ＥＣＣコードの付加や誤り訂正を行なって書換え保証回数を向上させるから、前記一部の記憶領域２０Ｂａには書換えの頻繁なパラメータデータ

等を格納すればよい。他の記憶領域２０Ｂｂには書換え頻度の低いプログラムデータなどを格納しても、当該他の記憶領域にはＥＣＣコードが付加されないから、記憶領域に拘わらず全てのライトデータに対して区別なくＥＣＣコードを付加する構成に比べて、実質的に無用のＥＣＣコードによる記憶領域の無駄な利用を省きながらフラッシュメモリセルの記憶情報の信頼性向上、換言すれば記憶情報の書換え保証回数の向上を実現することができる。

【００５６】

ライトデータに対するＥＣＣコードの付加、並びにＥＣＣコードによる誤り判定及び訂正の処理をＣＰＵ２によるプログラム実行で実現するから、ユーザデータビット数に対するＥＣＣコードのビット数の割合を定義するＥＣＣ方式はＥＣＣコード生成プログラム及びエラー訂正プログラムの記述内容で選択でき、フラッシュメモリのメモリセルのデバイス特性に合ったＥＣＣ方式の選択が容易である。例えば、図４に例示されるようにユーザデータビット数に対するＥＣＣコードのビット数が減るほど訂正能力は低くなるがＥＣＣコードによって費やされるメモリセルの割合（オーバヘッド）は少なくなる。したがって、ＥＣＣ処理のソフトウェアによりエラー訂正効率を容易に変えることができるから、マイクロコンピュータのデバイス能力に合ったエラー訂正方式を容易に選択することができ、これはＥＣＣコードのビット数から無駄を排除することを意味し、フラッシュメモリの記憶エリアの有効利用を保証する。要するに、ユーザデータに対するＥＣＣコードのオーバヘッドを小さくしてフラッシュメモリの利用効率を最大限とすることが可能になる。

【００５７】

《ＥＣＣ処理領域の規定》

前記ユーザ領域２０Ｂの一部の記憶領域２０Ｂａ をＥＣＣ処理対象とするとき、その記憶領域２０Ｂａを規定するのは、例えば記憶領域２０Ｂｂに格納されたユーザプログラムである。前記記憶領域２０Ｂａにパラメータを格納するとき、例えばユーザプログラムは、パラメータデータのソースアドレスと領域２０Ｂａのストア先アドレスを指定し、ＣＰＵの処理をＥＣＣコード生成プログラムにジャンプさせる。ＥＣＣコード生成プログラムが実行されることにより、ソースア

ドレスのパラメータデータに対してＥＣＣコードが生成され、パラメータデータとしてのユーザデータにＥＣＣコードを付加したＥＣＣ付きデータが前記ストア先アドレスに格納される。パラメータデータをリードするときは、ユーザプログラムは領域２０Ｂａのソースアドレスと任意のディスティネーションアドレスを指定し、ＣＰＵの処理をエラー判定プログラムにジャンプさる。エラー判定プログラムが実行されることにより、ソースアドレスのＥＣＣ付きデータがリードされ、これに対する誤り判定が行なわれ、必要な訂正が行なわれてパラメータデータとしてのユーザデータがディスティネーションアドレスにセットされる。ＥＣＣ処理を伴った上記パラメータデータの格納及びパラメータデータのリードにおいて、ユーザプログラムとＥＣＣコード生成プログラム及びエラー訂正プログラムとの間で行なわれるべきアクセスアドレス及びリードデータの受け渡しは、ＣＰＵ２の汎用レジスタ或いはＲＡＭ３の領域を介して行なわれる。この詳細は、ソフトウェアＥＣＣ処理の具体例で説明する。

《ユーザデータとＥＣＣコードとの対応》

上記処理において、ユーザデータとＥＣＣコードとの対応はユーザプログラム、ＥＣＣコード生成プログラム及びエラー訂正プログラムにおいて統一的に把握されていることが望ましい。例えば、図５に例示されるように、ＥＣＣコード生成対象とされるユーザデータの配列ＤＡ１に対し、ユーザデータと対応するＥＣＣコードを一つのレコードとするように一つの配列データＤＡ２としてフォーマット化し、或いは、ユーザデータの配列ＤＡ１に対してＥＣＣコードを別のデータ配列ＤＡ３として規定してよい。後者の場合は当然、ユーザデータの配列ＤＡ１とＥＣＣコードの配列ＤＡ３は先頭アドレスなどによって配列相互の対応付けが必要であることは言うまでもない。

【００５８】

図６にはユーザデータとＥＣＣコードとを一つの配列データ中に対応付けて領域２０Ｂａに書き込むときの処理手順が例示される。先ず、書き込み対象とされるユーザデータの配列が指定されてＥＣＣコード付きデータの書き込み処理が指示されると、ユーザデータの読み込みと展開が行なわれる（Ｓ１）。即ち、（Ｂ）に例示されるように、ユーザデータの配列ＤＡ１がワークメモリに読み込まれ

、そこで、読み込まれたユーザデータは規定のデータフォーマットにしたがって、（Ｂ）の配列ＥＸＴで例示されるように、ユーザデータの隣にＥＣＣコード領域を有するレコード形式に拡張される。次いで、（Ａ）に例示されるようにユーザデータに対してＥＣＣコードを生成し、これを対応レコードのＥＣＣコード領域にストアして、レコード配列ＤＡ２が形成される。このレコード配列ＤＡ２のデータに対して前記フラッシュメモリの領域２０Ｂａへの書き込みが行なわれる（Ｓ３～Ｓ７）。書き込み処理は、書き込みパルスの印加（Ｓ４）、書き込みデータのベリファイ（Ｓ５）、ベリファイ結果に基づく書き込み終了判定（Ｓ６）を行ない、所望の書き込み状態に到達していなければ再書き込みデータを演算してステップＳ４からの処理を繰り返し、規定回数繰り返してもステップＳ６の終了判定が満足できなければ書き込み異常終了とされ、ステップＳ６で規定の書き込み状態に到達すれば書き込み正常終了とされる。書き込みパルス印加の前には書き込みエリアに対する消去が終了されているものとする。

【００５９】

図７にはユーザデータとＥＣＣコードとを別の配列データとして対応付けて領域２０Ｂａに書き込むときの処理手順が例示される。先ず、書き込み対象とされるユーザデータの配列が指定されてＥＣＣコード付きデータの書き込み処理が指示されると、ユーザデータの読み込みが行なわれる（Ｓ１１）。即ち、（Ｂ）に例示されるように、ユーザデータの配列ＤＡ１がワークメモリに読み込まれる。次いで、（Ａ）に例示されるようにユーザデータに対してＥＣＣコードを生成し、これを別のデータ配列ＤＡ３としてストアする（Ｓ１２）。双方のデータ配列ＤＡ１，ＤＡ３に対して前記フラッシュメモリの領域２０Ｂａへの書き込みが行なわれる（Ｓ１３～Ｓ１７）。書き込み処理は前記ステップＳ４～Ｓ７の処置と同様である。

【００６０】

図８には図６で説明したレコード配列を有するＥＣＣコード付きデータをリードするときの処理手順が例示される。先ず、リード対象とされるＥＣＣコード付きデータのレコード先頭アドレスが設定され（Ｓ２１）、これに対応するＥＣＣコード付きデータがフラッシュメモリの領域２０Ｂａからワークメモリに読み込

まれる（Ｓ２２）。読み込まれたデータに対してエラー判定が行なわれる（Ｓ２３）。このとき、図６で説明したようにエラー判定の為に読み込まれた配列データは一定のフォーマット、即ち、ユーザデータとＥＣＣコードの所定ビット数単位のペアを単一レコードとする配列のフォーマットを有しているから、エラー訂正プログラムは、その規定のフォーマットであることを前提に、各レコードからユーザデータとＥＣＣコードを参照して誤りの判定を行なうことができる。訂正可能な誤りに対してはエラー訂正が実行され（Ｓ２４）、必要な訂正が実行されたリードデータがＲＡＭ３などの所定のエリアにストアされる（Ｓ２５）。

【００６１】

図９には図７で説明したユーザデータとＥＣＣコードが別々の配列データとされるＥＣＣコード付きデータをリードするときの処理手順が例示される。先ず、リード対象とされるＥＣＣコードの配列データの先頭アドレスとユーザデータの先頭アドレスが設定され（Ｓ３１）、これに対応するＥＣＣコードとユーザデータがフラッシュメモリの領域２０Ｂａからワークメモリに読み込まれ（Ｓ３２）、読み込まれたデータに対してエラー判定が行なわれる（Ｓ３３）。エラー判定において、対応するユーザデータとＥＣＣコードを参照する場合にも夫々の配列の構造を指定する情報がユーザプログラムを介して与えられなければならない。要するに、図７で説明したＥＣＣコード用配列データＤＡ３の生成に際して、当該配列データＤＡ３の先頭アドレス及び構造はユーザプログラムを介して与えられているから、その配列を利用する場合にも同じようにユーザプログラムから必要なアドレス情報及び配列の構造情報が与えられなければ、それを利用する事はできない。訂正可能な誤りに対してはエラー訂正が実行され（Ｓ３４）、必要な訂正が実行されたリードデータがＲＡＭ３などの所定のエリアにストアされる（Ｓ３５）。

【００６２】

図３の（Ｂ）で説明したように、前記ＥＣＣコード生成プログラム及びエラー訂正プログラムを前記書換え許容領域としてのユーザ領域２０Ｂに保持させれば、それらプログラムをユーザが開発して書き込むことができるという自由度を得る。したがって、ユーザデータとＥＣＣコードとのフォーマットについてはユー

ザ作成のプログラムに依存した任意フォーマットにする方がユーザにとって都合がよい場合もある。このとき、マイクロコンピュータ１のメーカーがそれらＥＣＣコード生成プログラム及びエラー訂正プログラムを提供しようとする場合にもユーザの使い勝手を考慮すれば、Ｃ言語のような高級言語で記載されたソースプログラムで与えるのがよい。そのような任意フォーマットを考慮したとき、ＥＣＣコードの付加及びＥＣＣコード付きデータを用いたエラー訂正処理方式には、例えば図７及び図９で説明した手順を採用してよい。

【００６３】

一方、図３の（Ｃ）で説明したように、消去禁止領域としてのブート領域２０Ａに前記ＥＣＣコード生成プログラム及びエラー訂正プログラムを格納することを想定する。この場合には、ブート領域２０Ａに一旦書き込まれたプログラムの書換えは原則不可能であるから、望ましくは、それらプログラムはマイクロコンピュータ１のメーカが書き込むのがよい。そうであるなら、ユーザデータとＥＣＣコードとのフォーマットに関しても、ユーザに対する自由度の保証という観点は低く、逆にユーザの負担軽減という観点より、固定フォーマットを採用する方が得策であると考えられる。前記ＥＣＣコード生成プログラムは、ＥＣＣコードを生成した後、生成したＥＣＣコードとそれに対応するデータとを規定のフォーマットに従いＥＣＣコード付加データとして前記不揮発性メモリの一部の記憶領域に格納し、エラー訂正プログラムは前記規定のフォーマットに従ってＥＣＣコード付加データを認識すればよい。そのような固定フォーマットを考慮したとき、ＥＣＣコードの付加及びＥＣＣコード付きデータを用いたエラー訂正処理方式として、例えば図６及び図８で説明した手順を採用すればよい。前記ＥＣＣコード生成プログラム及びエラー訂正プログラムを図３の（Ｄ）で説明したマスクＲＯＭ８に格納する場合も同様に考えてよい。

【００６４】

《オンボードプログラムモード》

オンボードプログラムモードについて説明する。この動作モードは、図３の（Ｂ）で説明したように前記ＥＣＣコード生成プログラム及びエラー訂正プログラム等をユーザ領域２０Ｂに保持させる時に必要な動作モードの一例である。

【００６５】

先ず、ブートモードによるオンボードプログラムの手順を説明する。パーソナルコンピュータ又はＥＰＲＯＭライタ等のホスト装置に書き込み制御プログラムと、前記ＥＣＣコード生成プログラム及びエラー訂正プログラムを用意し、Ｉ／Ｏ１１の所定のポートに接続する。所定の外部端子を規定の状態にしてマイクロコンピュータ１をブートモードに遷移させる。ブートモードに遷移すると、マイクロコンピュータ１はブート領域のブートプログラムを実行し、ＳＣＩ１２による通信を可能とし、ブート領域からＲＡＭ３に、フラッシュメモリ５のユーザ領域２０Ｂに対する消去プログラムと、ＳＣＩ１２を介する通信制御プログラムをロードする。ついで、ロードされた消去プログラムが実行されてユーザ領域２０Ｂが全面消去され、前記通信制御プログラムによってホストから書き込み制御プログラムがＲＡＭ３にロードされる。その後、書き込み制御プログラムが実行され、ホストが保有する前記ＥＣＣコード生成プログラム及びエラー訂正プログラム等がユーザ領域２０Ｂに書き込まれる。

【００６６】

次に、ユーザプログラムモードによるオンボードプログラムの手順を説明する。パーソナルコンピュータのようなホスト装置に書き込み・消去制御プログラムと、前記ＥＣＣコード生成プログラム及びエラー訂正プログラムを用意し、また、ユーザ領域２０Ｂにはホスト装置と間の転送制御プログラムを予め格納しておく。先ず、ＣＰＵ２は割込みに応答し或いはジャン命令を実行することにより前記転送制御プログラムを実行し、ホスト装置から書き込み・消去制御プログラムをＲＡＭ３に転送する。次に、ＲＡＭ３上で書き込み・消去制御プログラムを実行し、ユーザ領域２０Ｂの必要なエリアを消去し、そこに、ホストが保有する前記ＥＣＣコード生成プログラム及びエラー訂正プログラムを書き込む。

【００６７】

≪ＥＣＣ処理速度の向上≫

図１０には前記ＥＣＣコード生成プログラム及びエラー訂正プログラムの実行速度を向上させる為の一例が示される。即ち、前記フラッシュメモリ５のブート領域２０Ａ（図３の（Ｃ））又はユーザ領域２０Ｂ（図３の（Ｂ））から前記Ｅ

ＣＣコード生成プログラム及びエラー訂正プログラムをＲＡＭ３の所定のアドレス領域へ転送し、前記ＣＰＵ２には前記ＲＡＭ３に転送された前記ＥＣＣコード生成プログラム及びエラー訂正プログラムを実行させればよい。このとき、前記ＣＰＵ２はリセットの指示に応答して前記フラッシュメモリ５から前記ＲＡＭ３への転送制御プログラムを実行すればよい。そのような転送制御プログラムは例えばユーザ領域２０Ｂｂにおけるユーザプログラムとして、或いはブート領域２０Ａのプログラムとして所定のアドレス領域に格納しておけばよい。この方法は、ＲＡＭ３のアクセスサイクルがフラッシュメモリ５のアクセスサイクルより速い場合において、上記ＥＣＣコード生成プログラム及びエラー訂正プログラムのＣＰＵによる実行速度を高速にすることができる。

【００６８】

図１１にはエラー判定によるオーバヘッドを見掛け上解消するための一例が示される。即ち、前記ＣＰＵ２はリセットの指示に応答して、所定のリセット処理ユーザプログラムを実行し、前記フラッシュメモリ５の一部の記憶領域２０Ｂａから順次ＥＣＣコード付加パラメータデータＤＡＴＡ１を読み出し、読み出したＥＣＣコード付加パラメータデータＤＡＴＡ１に対し、前記エラー訂正プログラム２２の実行により前記エラー判定とエラーの訂正を行い、前記エラー判定とエラー訂正処理を経たパラメータデータＤＡＴＡ２を前記ＲＡＭ３に初期的にストアする。その後、ＣＰＵ２は制御用ユーザプログラムにしたがってＲＡＭ３から必要なデータをリードすればよく、リード動作に際してその都度エラー判定を行なうことを要しない。

【００６９】

なお、図１１の前記エラー訂正プログラム２２は、図１０で示されるように、フラッシュメモリ５の所定のアドレス領域からＲＡＭ３の所定のアドレス領域へ転送された状態を示しているしている。

【００７０】

図１２は、エラー判定によるオーバーヘッドを見掛け上解消する為の第２の方法の一例が示される。上記図１１は前記エラー訂正プログラム２２をＲＡＭ３のアドレス空間上でＣＰＵ２により実行する場合を示したが、図１２はＲＡＭ３の

リードアクセスサイクルがフラッシュメモリ５のリードアクセスサイクルより速くない（短くない）場合に有効な方法が示される。

【００７１】

フラッシュメモリ５のアクセスサイクルとＲＡＭ３のアクセルサイクルとが同一の場合、エラー訂正プログラム２２のＣＰＵ２による実行速度は、ＲＡＭ３の記憶領域上からエラー訂正プログラムをＣＰＵ２で実行しても、フラッシュメモリの記憶領域上からエラー訂正プログラムをＣＰＵ２で実行しても、その実行速度はそれほど変わらないと考えられる。

【００７２】

したがって、図１２に示されるように、フラッシュメモリ５の一部の記憶領域上にエラー訂正プログラム２２を格納し、そのプログラム２２をＲＡＭ３へ転送することなくフラッシュメモリ５の上記一部の記憶領域からＣＰＵ２で実行し、フラッシュメモリ５に一部の記憶領域２０Ｂａに格納されたＥＣＣコード付加パラメータＤＡＴＡ１１に対してエラー判定及びエラー訂正を行ない、エラー判定及びエラー訂正処理の施されたパラメータデータＤＡＴＡ２２をＲＡＭ３の所定の記憶領域に格納する。上記エラー訂正プログラムの処理は、上記図１１で説明されたように、フラッシュメモリ５の一部の領域２０Ｂｂに格納されたリセット処理ユーザープログラムが、リセットに応答するＣＰＵ２によって実行される事によって実行される。

【００７３】

《ソフトウェアＥＣＣ処理の原理的手法》

次にソフトウェアＥＣＣ処理の原理的な手法を説明する。ＥＣＣによる符号化や誤り訂正の公知の手法では例えばハミングコードの検査行列を用い、誤り訂正ではリードデータとハミングコードの行列演算を積和演算などを用いて行なうことができる。マイクロコンピュータ１におけるソフトウェアＥＣＣ処理では、データをｎビットとし、ｎビットのデータに対するＥＣＣコードをｍビットとするとき、ｍビットの相互に異なる２進数をｍ＋ｎ列に並べた行列テーブルを例えば図３の（Ｃ）に例示されるフラッシュメモリ５の記憶領域２０Ｂｂに固定データとして形成し、前記ＥＣＣコード生成プログラム及びエラー訂正プログラムの処

理にその行列テーブルを参照させる。

【００７４】

図１３には１６ビットデータに対する１ビット訂正のための行列テーブル４０の一例が示される。同図の行列テーブルは便宜上生成行列４１と検査行列４２に分けて図示されている。生成行列４１における行番号１～２１、列番号１～１６は、１～１６が１６ビットデータに関する番号、１７～２１が５ビットのＥＣＣコードに関する番号であると理解されたい。検査行列４２における行番号１～２１は生成行列と同意義、列番号１～５は５ビットのＥＣＣコードに関する番号であると理解されたい。

【００７５】

ＥＣＣコード付加データ、即ち符号語の生成は次のように行なう。例えば１６ビットのデータをＭ＝ｍ１５，ｍ１４，…，ｍ１，ｍ０に対して５ビットのＥＣＣコード（検査ビット）Ｐ＝ｐ４，ｐ３，ｐ２，ｐ１，ｐ０を生成して２１ビットの符号語を生成する。この符号語を生成するには生成行列４１とデータＭとの行列演算を行なえばよい。要するに、図１４のように生成行列４１とデータＭを並べ、データＭのビットが“１”に対応する生成行列４１の行を加算すれば良い。ここで行なう加算は２進数の加算であり、各ビット毎の排他的論理和（Ｅｘ－ＯＲ）を計算すればよい。この演算手法において、元の１６ビットのデータＭはそのまま残るから、生成行列４１におけるＥＣＣコード５ビット相当部分に関して排他的論理和を演算すればよい。例えば、Ｈ’８０４１（ｂ’１０００＿００００＿０１００＿０００１）の１６ビットデータに対してＥＣＣコードを生成するには、図１５に例示されるように、１６ビットデータＭに対して、前記生成行列４１の内、ＥＣＣコード５ビット相当部分に関する行列を抜き出した行列４１Ａを並べ、データの論理値“１”のビット位置に対応する前記行列テーブルの列の値を行方向のビット毎に排他的論理和を演算してＥＣＣコードを生成すればよい。図１６にはここで行う排他的論理和演算のための演算手段をＥＣＣコード１ビット分の構成を代表として例示する。図１６の構成では、２ビットの排他的論理和演算器４３の出力をディスティネーションレジスタ（Ｄｒｅｇ）４５が受け、排他的論理和演算器４３の一方の入力にはＤｒｅｇ４３の出力が帰還され、他

方の入力にはソースレジスタ（Ｓｒｅｇ）４５の出力が与えられる。前記排他的論理和演算においてＤｒｅｇ４３の初期値は“０”であり、Ｓｒｅｇ４３には順次、データの論理値“１”のビット位置に対応する前記行列テーブルの列の値が行方向のビット毎に入力され、排他的論理和演算器４３は双方の入力に対する排他的論理和演算結果をＤｒｅｇ４４にラッチし、これをＳｒｅｇ４３の次の出力との排他的論理和に用いる。最終的にＤｒｅｇ４４に得られた結果が、ＥＣＣコードの対応ビットの値になる。

【００７６】

尚、図１５、図１６の説明より明らかなように、生成行列４１は少なくとも図１４の列番号１７～２１の部分だけあればよい。

【００７７】

図１７には図１５の場合の排他的論理和演算結果が示される。同図のＸｅ－ＯＲ結果が演算されたＥＣＣコードである。Ｈ’８０４１の場合には、Ｍ＝ｂ’１０００＿０００＿０１００＿０００１にＰ＝ｂ’１１１０１を付加したデータＣ＝１０００＿０００＿０１００＿０００１，１１１０１が、２１ビットの符号語になる。

【００７８】

フラッシュメモリから読み出された符号語に対するエラー判定は、前記検査行列４２と２１ビットの符号語との積を計算する。実際に積を計算しようとすれば積和演算による行列演算を行なわなければならない。ここでは、図１８に例示されるように、検査行列４２に対して符号語Ｃを並べ、前記符号語の論理値“１”のビット位置に対応する前記検査行列４２の列の値を行方向のビット毎に排他的論理和を採る。この排他的論理和演算は図１６で説明したのと同じ手法で行えばよい。図１８の例はフラッシュメモリ５から読み出された符号語に誤りの無い場合を示しており、上記生成された符号語Ｃ＝１０００＿０００＿０１００＿０００１，１１１０１と同じ符号語が読み出された状態が示されている。

【００７９】

図１９には図１８の例において前記排他的論理和による演算結果が示される。図１９の結果の欄に示された５ビットの値Ｒｌｔが全ビット論理値“０”のとき

はエラー無と判定され、前記符号語Ｃに含まれる１６ビットのデータＭが正規データとされる。

【００８０】

図２０にはフラッシュメモリ５から読み出した符号語Ｃｅｒに１ビットの誤りを含む場合が例示される。図２０の結果の欄に示された５ビットの値Ｒｌｔが１ビットでも論理値“１”のときはエラー有りと判定される。誤りのあるビット位置は、前記検査行列４２の列から、前記排他的論理和によって得られた５ビットの２進数Ｒｌｔに一致する列に対応される位置の符号語ＣｅｒのビットＢｅｒである。この誤りビットＢｅｒに対し、その論理値を反転してエラー訂正を行えばよい。訂正された符号語に含まれる１６ビットのデータＭが、訂正後の正規データとされる。

【００８１】

図２０の説明より明らかなように訂正可能な１ビットの誤りを生じているとき、エラー判定の演算結果Ｒｌｔは検査行列４２の何れかの列のビットパターンに一致することになる。エラー判定の演算結果Ｒｌｔにそれ以外のビットパターンが現れたときは２ビット以上の訂正不能な誤りが発生していることになる。図２１にはそのような訂正不能を意味する１０種類のビットパターンが示される。検査行列４２によるこの例では、それら訂正不能を意味する１０種類のビットパターンの値は１０進数で２２以上の値にされるようになっているから、訂正不能の判定も容易である。

【００８２】

以上で説明したソフトウェアＥＣＣ処理では積和演算を要する行列演算を直接行わないから、ＣＰＵ２若しくはマイクロコンピュータ１は積和演算器を備えなくとも、ソフトウェアによるＥＣＣ処理を能率的に行なうことが可能になる。また、行列テーブルを用いるから、その都度生成行列や検査行列を生成しなくてもよい。

【００８３】

図２２には訂正不能エラーに対して例外処理を実行可能にする例が示される。前記エラー訂正プログラムの実行による前記エラー判定処理において訂正不能な

エラーが発生したき、誤動作防止の観点より、前記ＣＰＵ２は、前記エラー訂正プログラムの実行による前記エラー判定処理において訂正不能なエラーの発生を示す情報を外部から認識可能にＲＡＭ３のフラグ領域ＦＬＧ３０（或いはフラッシュコントロールモジュール６内のレジスタＦＭＬＣＲのリードエラービットＲＥＲ）に保持させる。例外処理プログラムのようなユーザプログラムを介してそのフラグ領域ＦＬＧを所定インターバル毎に参照させ、訂正不能なエラー発生を認識したとき、訂正不能なエラーを発生したデータブロック或いは全てのデータの書換え、即ち、新たなユーザデータに、ＥＣＣコードを生成し、これを付加して、領域２０Ｂａの所定エリアに書き込む処理を実行させる。

【００８４】

更にデータ破壊による誤動作防止を更に推し進める場合には、前記ＣＰＵ２は前記エラー訂正プログラムの実行によりエラー訂正可能なデータを検出した場合にも、それを示す情報を外部から認識可能にＲＡＭ３の所定記憶領域又は所定の汎用レジスタに保持させるとよい。そのような情報はワーニング情報として利用すればよく、例えば所定のユーザプログラムを介してその情報エリアを定期的に参照させればよい。これにより、ユーザシステムは、データが壊れかけている、ということを逸早く認識することができ、壊れかけているデータの書換え（再書込み）を促してデータの信頼性を更に向上させることが可能になる。要するに実際にデータエラーを生ずる事態の発生を未然に防止することができる。

【００８５】

《ＥＣＣコード生成処理の具体例》

ここでは図１７で説明した固定フォーマットのユーザデータを用意してＥＣＣコードを生成する処理を具体的に説明する。ＥＣＣコードの生成前に例えばユーザプログラムにしたがってユーザデータの読み込みとフォーマット展開が行なわれる。展開されたデータは図２３に例示されるように、ユーザデータ１６ビットに対して１６ビットの拡張ビットを割り当て、合計３２ビットが一つのＥＣＣデータブロックとして符号語領域になる。前記１６ビットの拡張ビットには５ビットのＥＣＣコード領域の他に１１ビットのワークビット領域が設けられているが、これは、必要なデータブロックをワード境界を単位にアクセスして得られるよ

うにする実用的な観点を考慮したものである。

【００８６】

図２４には一つのＥＣＣデータブロックの構成が更に詳細に示される。同図において符号化検査ビットとはＥＣＣコードを意味する。同図においてＥＣＣデータブロックのビット番号に対応して機能名が割当てられる。機能名Ｄ００～Ｄ１５はユーザデータであり、機能名Ｐ００～Ｐ０４はＥＣＣコードである。

【００８７】

図２５には行列テーブルのデータ（生成用データ）例が示される。同図に示される生成用データは前記機能名に対応され、夫々テーブル検索アドレス（検索アドレス）Ｘ～Ｘ＋１４が割当てられている。図２５の生成用データは、要するに図１８の検査行列４２と実質的に等しく、この検査行列４２の一部が図１５で説明した実質的な生成行列４１Ａになっている。図２５の機能名Ｄ００～Ｄ１５の生成用データは図１８の列番号１６～１のビット列に対応され、図２５の機能名Ｐ００～Ｐ０４の生成用データは図１８の列番号２１～１７のビット列に対応されている。図２５のテーブルを便宜上ＥＣＣテーブル（ＥＣＣＴＬＢ）とも記す。

【００８８】

図２６にはＥＣＣコード生成プログラムによる処理手順が例示される。ＥＣＣコード生成プログラムとそれに先立って実行されるユーザプログラムとの間のデータ及びアドレスの受け渡しにはＣＰＵ２の汎用レジスタ及びフラッシュコントロールモジュール６内のレジスタが利用される。即ち、ユーザプログラムは、フラッシュコントロールモジュール６内のレジスタＦＭＰＡＲ０に書き込みデータエリアの先頭アドレスをセットする。要するに、図２３に例示されるところのＲＡＭ３に展開されたデータの先頭アドレスがセットされる。汎用レジスタＲ０はユーザ書き込みデータレジスタ、Ｒ１は行列テーブル検索アドレスポインタ、Ｒ２は行列テーブル検索ストップ値、Ｒ３はユーザ書き込みデータ保存アドレスポインタ、Ｒ４はユーザ書き込みデータ保存アドレスストップ値、Ｒ５はビットマスク用データ、Ｒ６は検査ビット生成用変数、Ｒ７は論理値“１”のビットがあるかを検出するための変数、Ｒ８はビットに対応するテーブル値を格納する変数

、として利用される。

【００８９】

図２６において、先ず、レジスタＦＭＰＤＲ０の値をレジスタＲ３にセットしてＲＡＭ３上に展開されたユーザデータの先頭アドレスをストアする（Ｓ４０）。次に、レジスタＲ３の値にｈ’８０を加算してＲＡＭ上に展開されたユーザデータのストップアドレスをレジスタＲ４にストアする（Ｓ４１）。そして、レジスタＲ３の値がレジスタＲ４の値に到達するまで、以下の処理を行う（Ｓ４２）。すなわち、レジスタＲ３の値を利用して先頭１６ビットユーザデータをリードしてレジスタＲ０にロードする（Ｓ４３）。レジスタＲ１にはＥＣＣＴＬＢの先頭の検索アドレスＸをセットし、レジスタＲ２にはそのストップアドレスＸ＋ｈ’１０をセットする（Ｓ４４）。レジスタＲ５にビットマスクデータｈ’０００１をセットし、レジスタＲ６にデフォルトデータｈ’０００００をセットする（Ｓ４６）。そしてレジスタＲ１の値がレジスタＲ２の値に到達するまで次の処理を繰り返す（Ｓ４７）。要するに、レジスタＲ７にレジスタＲ５の値をロードし、レジスタＲ７とレジスタＲ０の値に対して論理積をとり、その結果をレジスタＲ７に返す（Ｓ４８）。レジスタＲ７の値が０より大きいかを判定する（Ｓ４９）。ステップＳ４９の判定結果が０より大きければ、その時のマスクデータＲ５の論理値“１”のビット位置と等しいビット位置にＲ０のユーザデータも論理値“１”のビットも持つことになる。その場合にはＥＣＣテーブルの生成用データ（Ｙ）をレジスタＲ８にストアし（Ｓ５０）、レジスタＲ６とＲ８の値に対して前記排他的論理和処理を行って、その値をレジスタスＲ６に返す。ステップＳ４９においてＲ７の値が０であれば、レジスタＲ５の値を１ビット左シフトしてマスクビット位置を次のビット位置とし（Ｓ５２）、レジスタＲ１にｈ’０１を加算して、ＥＣＣＴＬＢの検索アドレスを次アドレスに進める（Ｓ５３）。そして再度ステップＳ４８の処理に戻る。ステップＳ４８～Ｓ５３の処理はＥＣＣＴＬＢの検索アドレスがストップアドレスになるまで繰り返され（Ｓ４７）、ここまでの処理により、図１７で説明したようなＥＣＣコードＰがレジスタＲ６に保持される。そして、次にレジスタＲ３の値を１ワード分だけインクリメントし（Ｓ５４）、レジスタＲ３の値が指すアドレス（ＲＡＭ３上に展開された図２３のデー

タフォーマットの拡張領域）に、レジスタＲ６が保有するＥＣＣコードをストアする（Ｓ５５）。そして次のユーザデータに対して同様の処理を行う為にレジスタＲ３のアドレスを１ワード分インクリメントする。上記ステップＳ４３～Ｓ５６までの処理をＲ３＜Ｒ４になるまで繰り返すことによって、ＲＡＭ３上に展開された一群のユーザデータに対してＥＣＣコードが付加される。この後、ＥＣＣコード生成プログラムの実行を終えて、直前のユーザプログラムにリターンする。特に図示はしないが、リターンされたユーザプログラムは、ＲＡＭ３上に展開されたＥＣＣコード付加データを、フラッシュメモリ５の所定の領域２０Ｂａに書き込む。

【００９０】

《エラー判定処理の具体例》

図２７には図２６の手順で生成されたＥＣＣコード付加データをリードしたときのエラー判定処理の詳細が示される。エラー判定処理プログラムとそれに先立って実行されるユーザプログラムとの間のデータ及びアドレスの受け渡しにはＣＰＵ２の汎用レジスタ及びフラッシュコントロールモジュール６内のレジスタが利用される。即ち、ユーザプログラムは、フラッシュコントロールモジュール６内のレジスタＦＭＰＡＲ０にユーザリードアドレスをセットする。要するに、図３の領域２０Ｂａ中のリードアドレスがセットされる。汎用レジスタＲ０はエラー訂正前のリードアドレス、Ｒ１は検査ビット保存変数、Ｒ２はエラー検出用レジスタ、Ｒ３はユーザリードアドレス、Ｒ４はＥＣＣ計算用中間テーブル、Ｒ５はＥＣＣＴＬＢ検索アドレス、Ｒ６はＥＣＣＴＬＢ検索ストップアドレス、Ｒ７はビットマスクデータ、Ｒ８は論理値“１”検出用変数、Ｒ９はＥＣＣＴＬＢデータ保存用変数、として利用される。

【００９１】

図２７において、先ず、レジスタＦＭＰＡＲ０のユーザリードアドレスをレジスタＲ３にセットし（Ｓ６０）、このレジスタＲ３のリードアドレスを利用してユーザデータをレジスタＲ０にストアする（Ｓ６１）。更にアドレスを１ワード分インクリメントし（Ｓ６２）、後続のＥＣＣコードのデータをレジスタＲ１にストアし、且つレジスタＲ１のデータをレジスタＲ４にコピーする（Ｓ６３）。

前記レジスタＲ４を左に１６ビットシフトを行い、そのシフト結果に対してｈ’００１Ｆ０００００との論理積を採る。これによってレジスタＲ４には、その第１７ビット目から第２１ビットにＥＣＣコードが配置され、その他のビットを“０”としたデータを得る（Ｓ６４）。更に、レジスタＲ０の値に対してｈ’０００００ＦＦＦＦとの論理積を採り、その結果をレジスタＲ０に返し、レジスタＲ４の値にレジスタレジスタＲ０の値を加えてその演算結果をレジスタＲ４に返す（Ｓ６５）。これにより図示の“１”ビット検索テーブルがレジスタＲ４に得られる。“１”ビット検索テーブルにおける未使用エリアは論理値“０”になっている。次にレジスタＲ２を論理値“０”に初期化し（Ｓ６６）、レジスタＲ５にＥＣＣＴＬＢの検索アドレスＸをセットし、レジスタＲ６に検索ストップアドレスＸ＋ｈ’１６をセットする（Ｓ６７）。レジスタＲ７にはビットマスクデータｈ’００００００００１をセットする（Ｓ６８）。そしてＲ５の値がＲ６の値になるまで以下の処理を繰り返す。即ち、レジスタＲ８にレジスタＲ４の値をセットし、Ｒ７とＲ８の論理積をＲ８に返し、Ｒ８が０より大きいかを判定する（Ｓ７１）。ステップＳ７１の判定結果が０より大きければ、その時のＲ７のマスクデータの論理値“１”のビット位置と等しいビット位置にＲ４の符号化データも論理値“１”のビットも持つことになる。その場合にはＥＣＣテーブルＥＣＣＴＬＢの生成用データ（Ｙ）をレジスタＲ９にストアし（Ｓ７２）、レジスタＲ２とＲ９の値に対して前記排他的論理和処理を行って、その値をレジスタスＲ２に返す（Ｓ７３）。ステップＳ７１においてＲ８の値が０であれば、レジスタＲ７の値を１ビット左シフトしてマスクビット位置を次のビット位置とし（Ｓ７４）、レジスタＲ５にｈ’０１を加算して、ＥＣＣＴＬＢの検索アドレスを次アドレスに進める（Ｓ７５）。そして再度ステップＳ７０の処理に戻る。ステップＳ７０～Ｓ７７の処理はＥＣＣＴＬＢの検索アドレスがストップアドレスになるまで繰り返され（Ｓ６９）、ここまでの処理により、図１９及び図２０で説明したような判定結果Ｒｌｔがレジスタ Ｒ２に保持される。そして、レジスタＲ２の値が全ビット“０”か否かを判定し（Ｓ７６）、そうでなければエラーが有るので、訂正可能な１ビットエラーに対して対してはサブルーチンとして後述のエラー訂正処理を実行する。そして、レジスタＦＭＰＤＲ０が示す領域に、レジスタＲ４の下

位１６ビットの値をストアし（Ｓ７）、レジスタＦＰＦＲにパス情報をセットする。尚、特に図示はしないがステップＳ７６ではエラーが訂正不能であるか否かも判定し、訂正不能であれば図２２で説明した訂正不能エラー発生の通知処理を行って例外処理を待ってもよい。

【００９２】

《エラー訂正処理の具体例》

図２８には１ビットの誤りに対するエラー訂正処理の詳細が示される。エラー訂正処理プログラムにおけるＣＰＵ２の汎用レジスタ及びフラッシュコントロールモジュール６内のレジスタの利用形態はエラー判定処理の場合と同じである。図２７において、先ず、レジスタＲ５にＥＣＣＴＬＢの先頭検索アドレスＸをセットし、レジスタＲ６にＥＣＣＴＬＢの検索ストップアドレスをセットする（Ｓ８０）。レジスタＲ７にビット反転用マスクデータｈ’０００１をセットする（Ｓ８１）。そしてＲ５の値がＲ６の値になるまで以下の処理を繰り返す。即ち、レジスタＲ５の検索アドレスに対応する生成用データをＥＣＣＴＬＢからレジスタＲ９にストアする。そして、Ｒ９の値が前記レジスタＲ２の判定結果Ｒ１ｔに一致するかが判別され、一致していれば、その時のＲ７のビット反転用マスクデータにおける“１”のビット位置に対応するユーザデータのビット位置に誤りがある。前記エラー判定処理のステップＳ７７で説明したようにレジスタＲ４の下位１６ビットを切出して、正規のユーザデータとするから、そのとき、レジスタＲ７のマスクデータとレジスタＲ４のデータとを対応ビット毎に排他的論理和を採ることによって、マスクデータのビット“１”に対応するユーザデータのビットだけが論理値反転され、エラー訂正されたユーザデータがレジスタＲ４に返される（Ｓ８５）。ステップＳ８４においてＲ９の値がＲ２の値に一致しなければ、レジスタＲ５にｈ’０１を加算して、ＥＣＣＴＬＢの検索アドレスを次アドレスに進め（Ｓ８６）、レジスタＲ７のマスクデータを１ビットシフトし（Ｓ８７）、再度ステップＳ８３の処理に戻る。ステップＳ８３～Ｓ８７の処理はＥＣＣＴＬＢの検索アドレスがストップアドレスになるまで繰り返されて（Ｓ８２）、当該サブルーチンを終了する。このサブルーチンが終了されたとき、レジスタＲ４には誤り訂正された正規のユーザデータを含んでいる。

【００９３】

《マルチチップのデータ処理システム》

図２９にはマルチチップのデータ処理システムが例示される。同図に示されるデータ処理システムはデータプロセッサ５０とフラッシュッメモリ５１が夫々別々に半導体集積回路化されてバス５２で接続され、その他に、単一又は複数個の半導体集積回路で構成された周辺回路５３がバス５２に接続される。データプロセッサ５０はＣＰＵ６５、ＲＡＭ６６、ＲＯＭ６７、及びＩ／Ｏ６８を有する。フラッシュメモリ５１は汎用フラッシュメモリであり、フラッシュメモリセルがマトリクス配置されたメモリセルアレイ７０、ＸＤＥ・ＤＲＶ７１、ＹＤＥ７２、ＴＧＮ７３、ＶＧＮ７４、ＹＳＷ７５、ＳＡＡ７６、書き込み・消去制御回路７７を有する。フラッシュッメモリ５１の基本的な構成は前記フラッシュメモリ５と同様であるからその詳細な説明は省略するが、書き込み及び消去を制御するロジック回路として書き込み・消去制御回路７７を専用に備える。フラッシュメモリ５１の動作はＣＰＵ６５から与えられるコマンド並びにアクセス制御信号によって決定される。ＣＰＵ６５はフラッシュメモリ５１の位置部の記憶領域７０Ｅを頻繁に書き換えるパラメータデータのようなデータの記憶領域として用いる。前述と同様にその領域７０Ｅは他の領域よりも書き換え保証回数を向上させる為にＥＣＣ処理の対象とされる。ＲＯＭ６７は前記領域７０Ｅをアクセスしてデータ処理を行うユーザプログラム６７Ｐを保有する。このユーザプログラム６７Ｐは、前記領域７０Ｅにパラメータデータを書き込むときに実行されるＥＣＣコード生成プログラム６７Ｐ１、前記領域７０Ｅから読み出したＥＣＣコード付加データに対するエラー判定及び誤り訂正を行う為のエラー訂正プログラム６７Ｐ２を有する。前記ＥＣＣコード生成プログラム及びエラー訂正プログラムは所定のユーザプログラムから呼び出されて実行される。

【００９４】

上記マルチチップのデータ処理システムにおいても図１のシングルチップのマイクロコンピュータ化されたデータ処理システムと同様に、ＥＣＣコードによる記憶領域の利用の無駄を省いて記憶情報の信頼性を向上させることができ、ＥＣＣコードによる記憶領域の利用の無駄を省いて記憶情報の書換え保証回数を向上

させることが可能であり、デバイスの特性に合ったＥＣＣ方式を選択してデータに対するＥＣＣコードのオーバヘッドを小さくし記憶領域の利用効率を最大限とすることが可能である等の効果を得ることができる。

【００９５】

《多値フラッシュメモリへの考慮》

前記フラッシュメモリ５，５１は、１個のフラッシュッメモリセルに２ビット以上の記憶情報を保持させることが可能な多値フラッシュメモリであってもよい。すなわち、１個のフラッシュメモリセルは、情報記憶に際して複数ビットの書き込みデータで指定される４種類以上の閾値電圧の中の一つの閾値電圧に設定され、情報読み出しに際して閾値電圧の状態を対応する複数ビットの記憶情報として出力する、１個のフラッシュメモリセルの記憶情報を複数ビット化したメモリである。ここでは、一つのフラッシュメモリセルに２ビットの情報を書き込むことができ、かつその情報を読み出すことができるフラッシュメモリを一例とする。このようなフラッシュメモリが実現しようとする多値情報記憶技術において、一つのメモリセルの情報記憶状態は、例えば消去状態（“１１”）、第１の書き込み状態（“１０”）、第２の書き込み状態（“００”）、第３の書き込み状態（“０１”）の中から選ばれた一つの状態とされる。全部で４通りの情報記憶状態は、２ビットのデータによって決定される状態とされる。即ち、２ビットのデータを一つのメモリセルで記憶する。この４値のデータと閾値電圧との関係は、図３０の閾値電圧分布図に示される通りである。フラッシュメモリセルの記憶データの値と閾値電圧との関係を図３０のように規定すると、情報記憶後に、閾値電圧が不所望に変化しても隣の閾値電圧のデータとは相互に１ビットしか相違しないようになる。したがって、データエラーを生じても殆どが１ビットエラーになり、１ビットエラーに対して訂正可能なＥＣＣ処理によるデータの信頼性を高く維持することが可能になる。換言すれば、多値フラッシュメモリセルの閾値電圧が近いところをデータハミング距離が１になるようにすれば、相対的に少ないビット数のＥＣＣコードで高い信頼性を得ることができ、書換え保証回数の向上が容易になる。

【００９６】

以上本発明者によってなされた発明を実施形態に基づいて具体的に説明したが、本発明はそれに限定されるものではなく、その要旨を逸脱しない範囲において種々変更可能であることは言うまでもない。

【００９７】

例えば、フラッシュメモリセルはフローティングゲートとコントロールゲートの縦積み構造に限定されず、ＭＯＳトランジスタのゲート電極をフローティングゲート電極とし当該ゲート電極を延在させて形成したＭＯＳゲート容量を介してチャネル領域をコントロールゲートに用いるようなデバイス構造などを採用してもよい。また、不揮発性記憶素子はフラッシュメモリに限定されず、ＭＮＯＳ（メタル・ナイトライド・オキサイド・セミコンダクタ）トランジスタを記憶素子とするＥＥＰＲＯＭ（エレクトリカリ・イレーザブル・アンド・プログラマブル・リード･オンリ･メモリ）のような不揮発性メモリ、或いは誘電体メモリ等であってもよい。

【００９８】

データ処理システムは、１個の半導体チップに前記不揮発性メモリ及び中央処理装置が形成されたシングルチップのマイクロコンピュータとして実現することが可能であり、その一方において、前述の如く前記データ処理システムは、前記不揮発性メモリ及び中央処理装置が夫々別々の半導体チップに形成されたマルチチップ形態で実現してもよい。そして、データ処理システムは、シングルチップ及びマルチチップのマイクロコンピュータに限定されず、フラッシュメモリを内蔵したグラフィックスコントローラ、ＤＲＡＭを専用ロジック回路と共に混載したシステムＬＳＩ、その他のマルチチップによる電子回路に広く適用する事ができる。

【００９９】

【発明の効果】

本願において開示される発明のうち代表的なものによって得られる効果を簡単に説明すれば下記の通りである。

【０１００】

すなわち、一部の記憶領域にだけＥＣＣコードの付加や誤り訂正を行なって書

換え保証回数を向上させるから、記憶領域に拘わらず全てのライトデータに対して区別なくＥＣＣコードを付加する構成に比べて、実質的に無用のＥＣＣコードによる記憶領域の無駄を省くことができる。更に、ＥＣＣ処理をソフトウェアで対処するから不揮発性メモリのデバイス特性に合ったＥＣＣ訂正能力を容易に選択することができる。このように、データ処理システムにおいてＥＣＣコードによる記憶領域の利用の無駄を省いて不揮発性メモリに記憶された情報の信頼性を向上させることができ、また、ＥＣＣコードによる記憶領域の利用の無駄を省いて不揮発性メモリにおける記憶情報の書換え保証回数を向上させることができる。更に、デバイスの特性に合ったＥＣＣ方式を選択でき、データに対するＥＣＣコードのオーバヘッドを小さくして記憶領域の利用効率を最大限とすることが可能である。

【０１０１】

ＥＣＣコードが付加されたデータに対する誤り判定及び訂正の処理によるデータリード動作の遅延を抑えることが可能である。

【０１０２】

積和演算などを行うことなくＥＣＣコードの生成を効率的に行なうことができ、ＥＣＣコードが付加されたデータの誤り判定を効率的に行なうことができる。

【図面の簡単な説明】

【図１】

本発明の一例に係るシングルチップのマイクロコンピュータのブロック図である。

【図２】

ＣＰＵの具体例を示すブロック図である。

【図３】

書換え保証回数を向上させる記憶領域、ＥＣＣコード生成プログラム及びエラー訂正プログラムの記憶領域のマッピング例を示す説明図である。

【図４】

ユーザデータのビット数に対するＥＣＣコードのビット数による誤り訂正能力とオーバーヘッドとの関係を例示する説明図である。

【図５】

ユーザデータとＥＣＣコードを一つの配列データ中に対応付けての配列するデータフォーマットと、ユーザデータとＥＣＣコードとを別の配列データとして対応付けるデータフォーマットとを例示する説明図である。

【図６】

ユーザデータとＥＣＣコードとを一つの配列データ中に対応付けて書き込むときの処理手順を例示する説明図である。

【図７】

ユーザデータとＥＣＣコードとを別の配列データとして対応付けて書き込むときの処理手順を例示する説明図である。

【図８】

図６で説明したレコード配列を有するＥＣＣコード付きデータをリードするときの処理手順を例示する説明図である。

【図９】

図７で説明したユーザデータとＥＣＣコードが別々の配列データとされるＥＣＣコード付きデータをリードするときの処理手順を例示する説明図である。

【図１０】

ＥＣＣコード生成プログラム及びエラー訂正プログラムの実行速度を向上させる為の一例を示す説明図である。

【図１１】

エラー判定によるオーバヘッドを見掛け上解消するための一例を示す説明図である。

【図１２】

エラー判定によるオーバヘッドを見掛け上解消するための別の一例を示す説明図である。

【図１３】

１６ビットデータに対する１ビット訂正のための行列テーブル４０の一例を示す説明図である。

【図１４】

ＥＣＣコードを付加した符号語を生成する原理的手法の説明図である。

【図１５】

H’８０４１の１６ビットデータに対し排他的論理和を用いてＥＣＣコードを生成する具体例の説明図である。

【図１６】

排他的論理和演算のための演算手段をＥＣＣコード１ビット分の構成を代表として例示するブロック図である。

【図１７】

図１５の例において排他的論理和を採った具体的な演算結果であるＥＣＣコードを例示する説明図である。

【図１８】

エラー判定処理の原理的手法を示す説明図である。

【図１９】

図１８の例において前記排他的論理和による誤り無の判定結果が例示される説明図である。

【図２０】

図１９に対して１ビット誤りの有る判定結果が得られる場合を例示する説明図である。

【図２１】

２ビット以上の訂正不能な誤りが発生している時の判定結果を列挙する説明図である。

【図２２】

訂正不能エラーに対して例外処理を実行可能にする例を示す説明図である。

【図２３】

ユーザデータとＥＣＣコードの符号語に採用される固定フォーマットの一例を示す説明図である。

【図２４】

図２３の固定フォーマットの更に詳細を例示する説明図である。

【図２５】

行列テーブルのデータを保有するＥＣＣテーブルを例示する説明図である。

【図２６】

ＥＣＣコード生成プログラムによる処理手順を例示するフローチャートである。

【図２７】

図２６の手順で生成されたＥＣＣコード付加データをリードしたときのエラー判定処理の詳細を示すフローチャートである。

【図２８】

１ビットの誤りに対するエラー訂正処理の詳細を示すフローチャートである。

【図２９】

マルチチップのデータ処理システムを例示するブロック図である。

【図３０】

多値フラッシュメモリにおける４値のデータと閾値電圧との関係を例示する説明図である。

【符号の説明】

１　マイクロコンピュータ

２　ＣＰＵ

３　ＲＡＭ

５　フラッシュメモリ

６　フラッシュコントロールモジュール

８　マスクＲＯＭ

２０　メモリセルアレイ

２０Ａ　ブート領域

２０Ｂ　ユーザ領域

２０Ｂａ　相対的に書換え保証回数の高い記憶領域

２０Ｂｂ　相対的に書換え保証回数の低い記憶領域

２１　ＥＣＣコード生成プログラム

２２　エラー訂正プログラム

ＤＡ１　ユーザデータの配列

ＤＡ２　固定フォーマットの符号語の配列

ＤＡ３　ＥＣＣコードのみから成るデータ配列

ＤＡＴＡ１　ＥＣＣコード付加パラメータデータ

ＤＡＴＡ２　エラー判定・訂正済パラメータデータ

ＲＥＲ　リードエラービット

３０　リードエラーフラグ

４０　行列テーブル

４１　生成行列

４２　検査行列

Ｍ　データ（ユーザデータ）

４１Ａ　生成行列の内ＥＣＣに関する部分を抜き出した行列

Ｐ　ＥＣＣコード

Ｃ　符号語

Ｒｌｔ　エラー判定の演算結果

Ｃｅｒ　１ビットの誤りを含む符号語

Ｂｅｒ　誤りビット

５０　データプロセッサ

５１　フラッシュメモリ

６５　ＣＰＵ

６６　ＲＡＭ

６７　ＲＯＭ

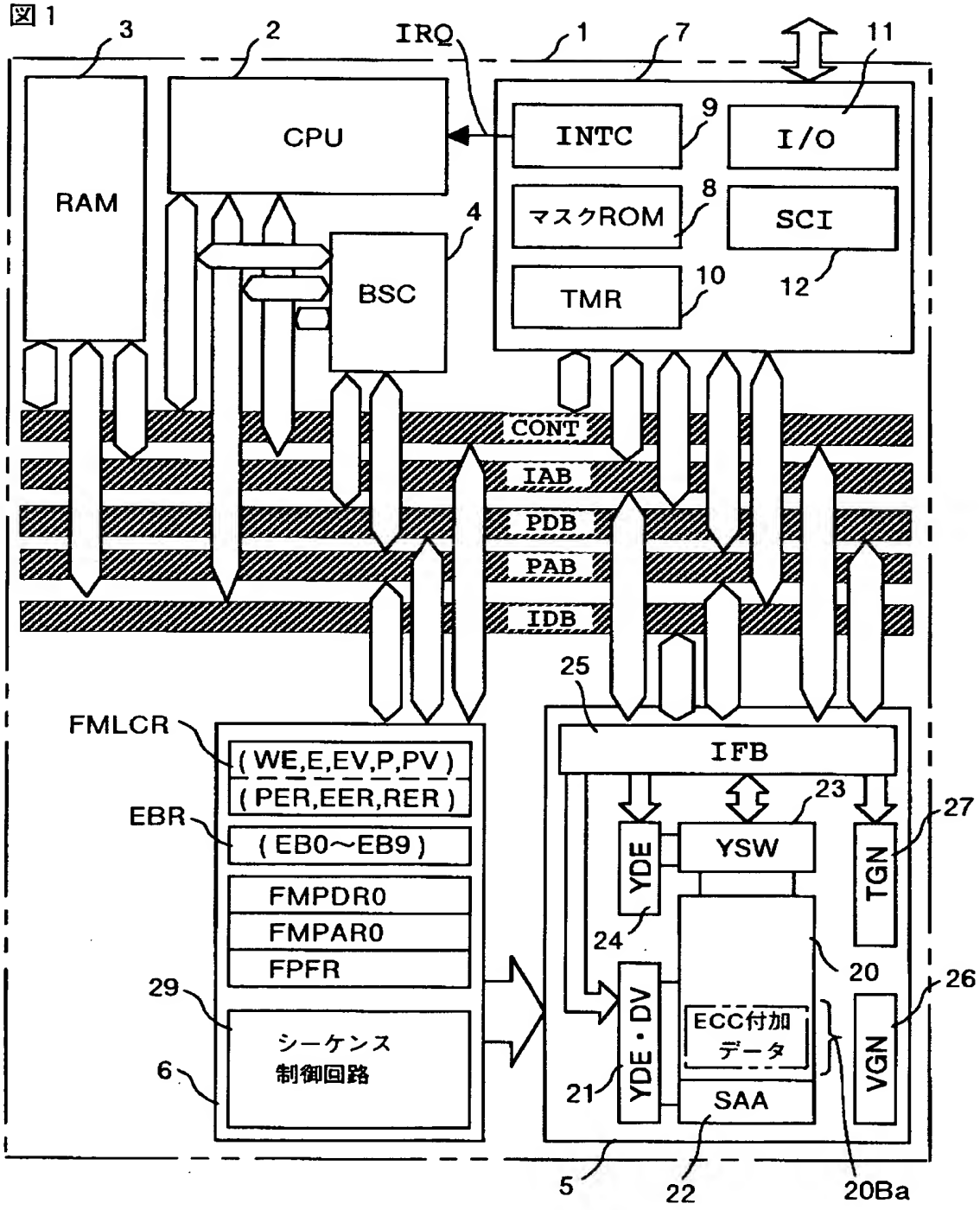
７０　メモリセルアレイ

７０Ｅ　相対的に書換え保証回数の高い記憶領域

【書類名】　図面

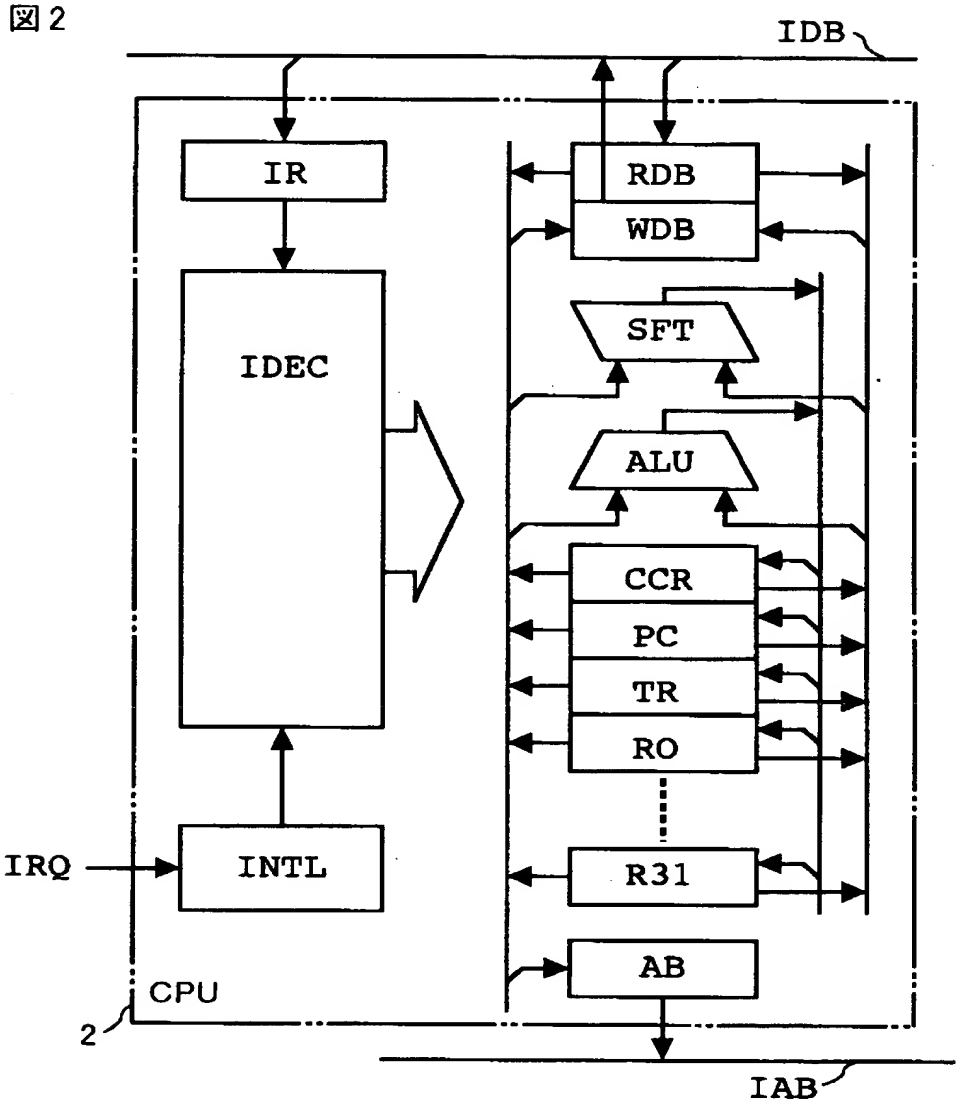
【図１】

図１


3
2
IRQ
1
7
11
RAM
CPU
INTC
9
I/O
マスクROM
8
SCI
4
BSC
TMR
10
12
CONT
IAB
PDB
PAB
IDB
25
FMLCR
(WE,E,EV,P,PV)
(PER,EER,RER)
EBR
(EB0～EB9)
FMPDR0
FMPAR0
FPFR
IFB
23
27
YDE
YSW
TGN
24
20
26
29
シーケンス
制御回路
6
YDE・DV
ECC付加
データ
21
SAA
VGN
5
22
20Ba

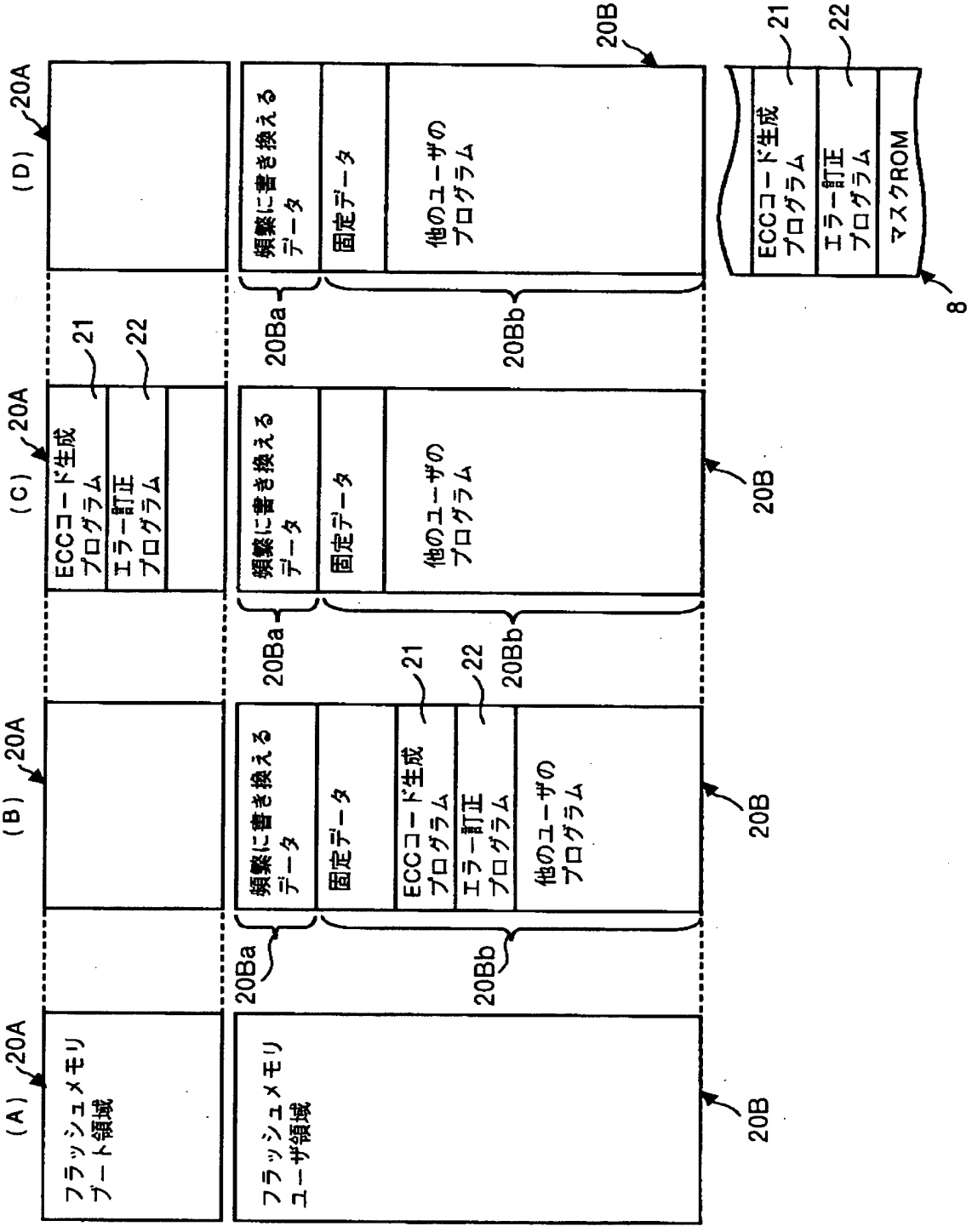
【図２】

図２



【図３】

図３


(A)
(B)
(C)
(D)
20A
フラッシュメモリ
ブート領域
フラッシュメモリ
ユーザ領域
20B
20Ba
20Bb
頻繁に書き換える
データ
固定データ
ECCコード生成
プログラム
エラー訂正
プログラム
他のユーザの
プログラム
21
22
マスクROM
8

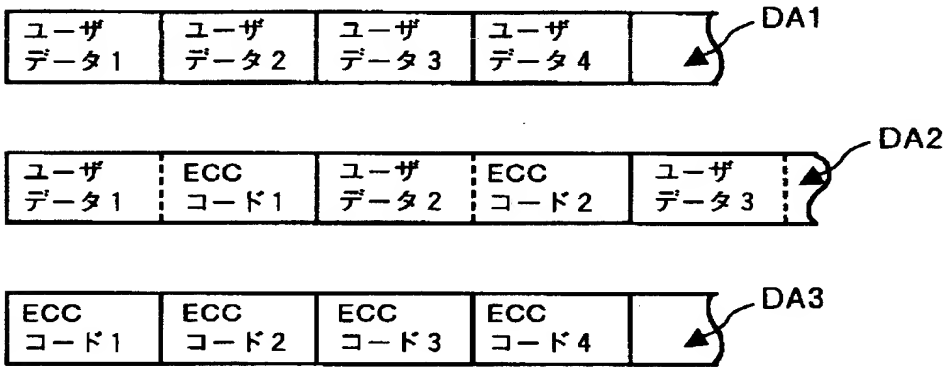
【図４】

図４

ユーザデータビット数	ECCコードビット数	オーバヘッド	訂正能力
8ビット	4ビット	50%	8ビット中1ビット
16ビット	5ビット	31%	16ビット中1ビット
32ビット	6ビット	19%	32ビット中1ビット
64ビット	7ビット	11%	64ビット中1ビット
128ビット	8ビット	6%	128ビット中1ビット
256ビット	9ビット	4%	256ビット中1ビット

【図５】

図５



【図６】

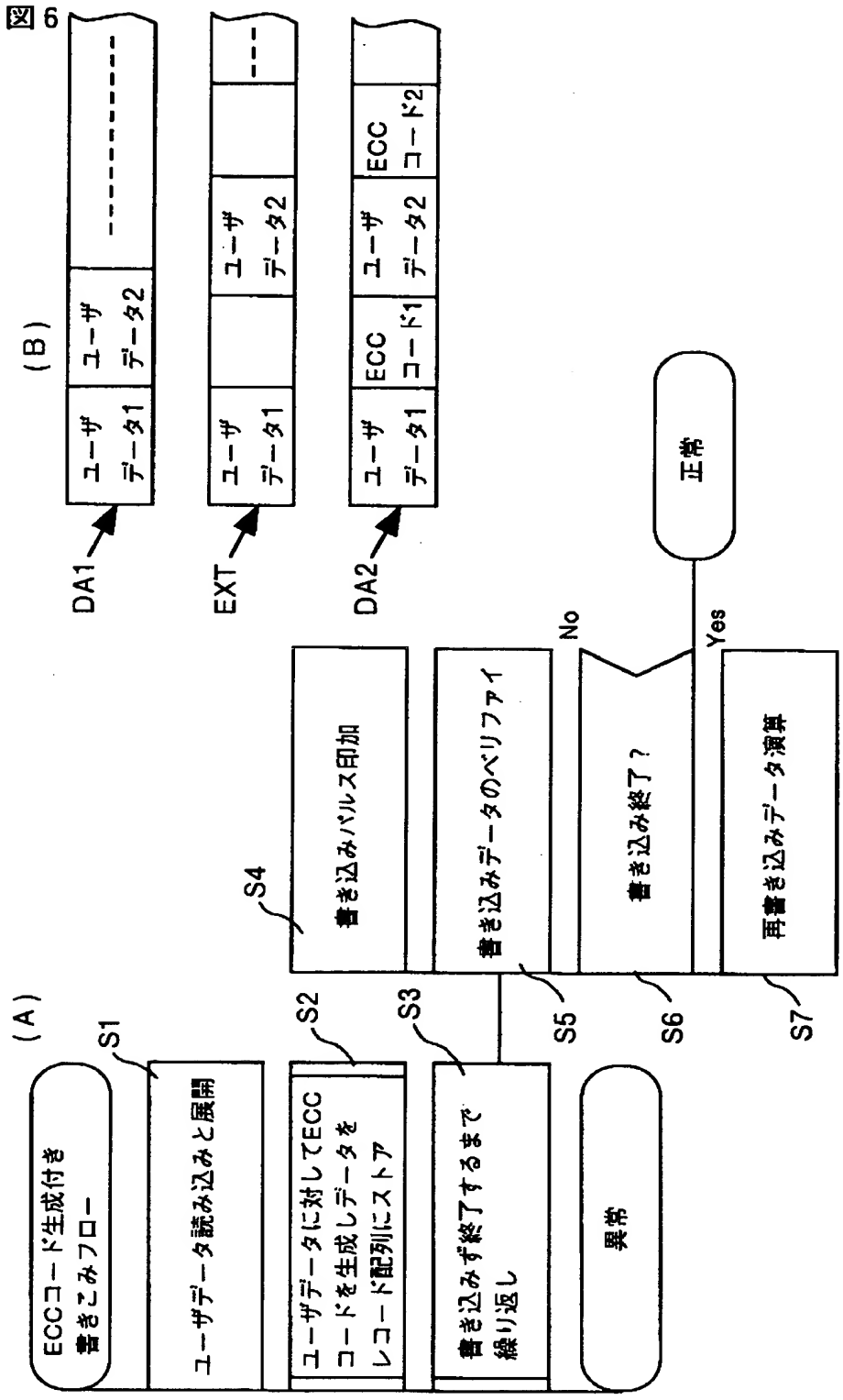
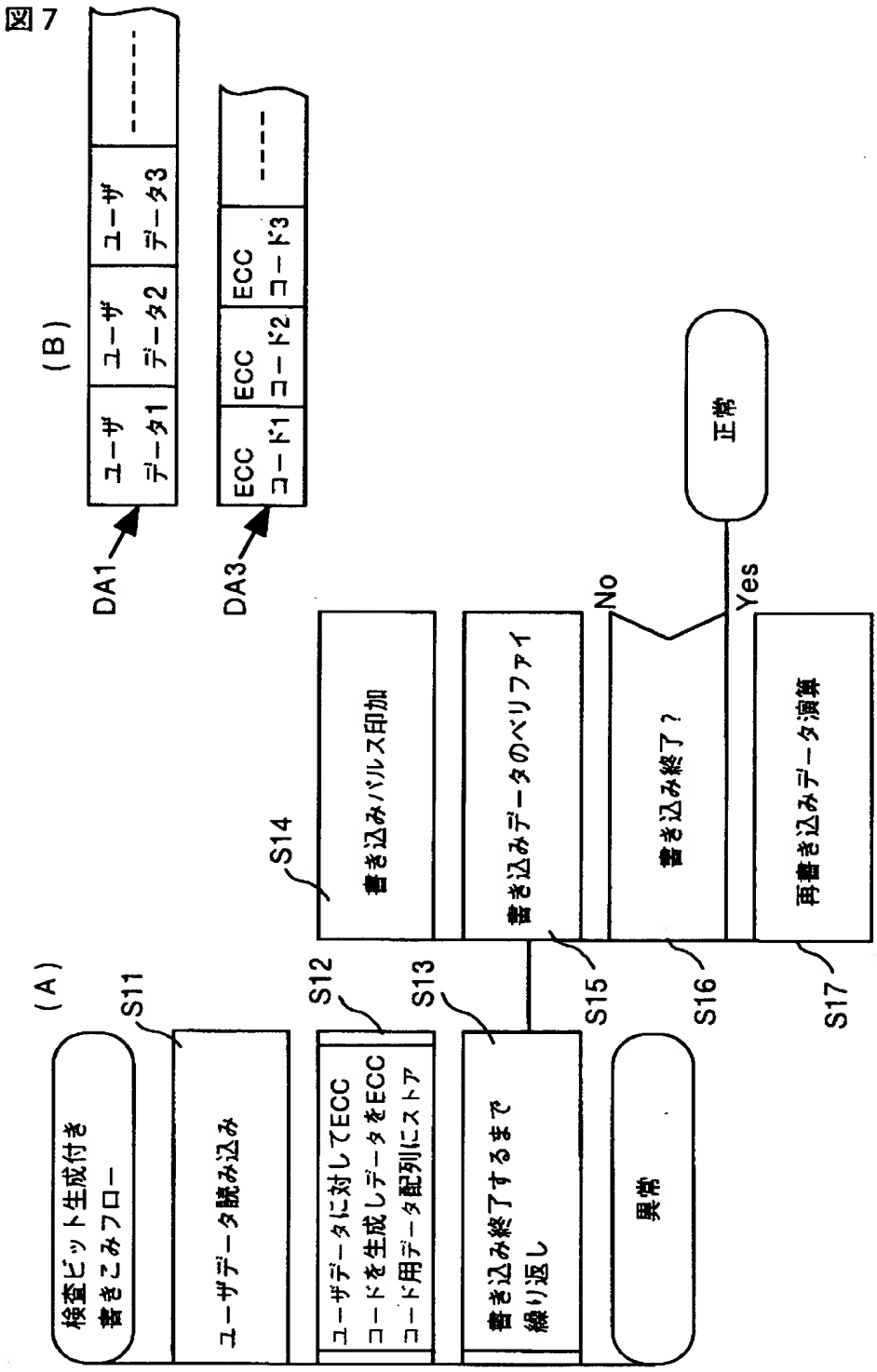


図６
(B)
ユーザデータ１
ユーザデータ２
DA1
EXT
DA2
ECCコード１
ECCコード２
(A)
ECCコード生成付き書きこみフロー
ユーザデータ読み込みと展開
ユーザデータに対してECCコードを生成しデータをレコード配列にストア
書き込み完了するまで繰り返し
異常
書き込みパルス印加
書き込みデータのベリファイ
書き込み終了？
No
Yes
正常
再書き込みデータ演算
S1
S2
S3
S4
S5
S6
S7

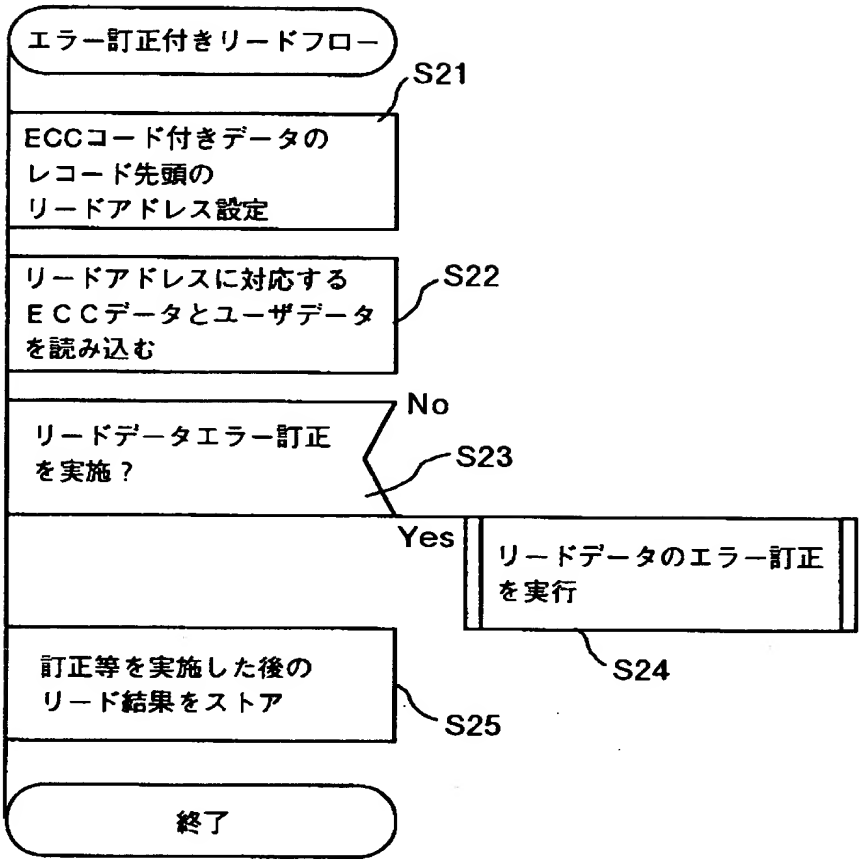
【図７】

図７


(A)
検査ビット生成付き
書きこみフロー
S11
ユーザデータ読み込み
S12
ユーザデータに対してECC
コードを生成しデータをECC
コード用データ配列にストア
S13
書き込み終了するまで
繰り返し
異常
S14
書き込みパルス印加
S15
書き込みデータのベリファイ
S16
No
書き込み終了？
Yes
正常
S17
再書き込みデータ演算
(B)
DA1
ユーザ
データ1
ユーザ
データ2
ユーザ
データ3
DA3
ECC
コード1
ECC
コード2
ECC
コード3

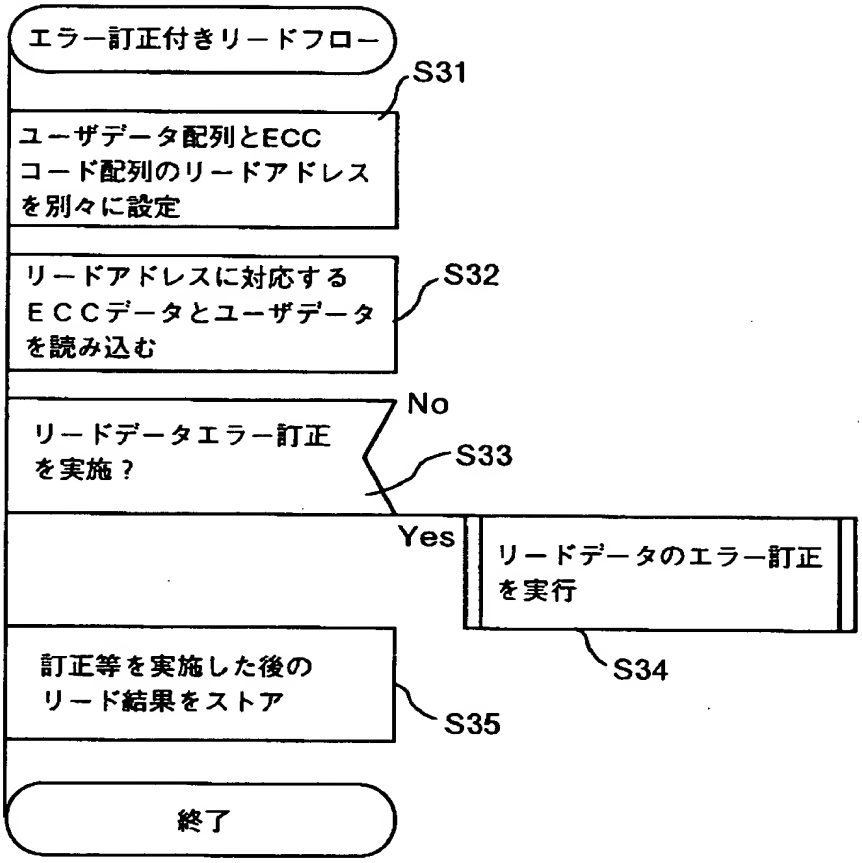
【図８】

図８


エラー訂正付きリードフロー
S21
ECCコード付きデータの
レコード先頭の
リードアドレス設定
リードアドレスに対応する
ＥＣＣデータとユーザデータ
を読み込む
S22
No
リードデータエラー訂正
を実施？
S23
Yes
リードデータのエラー訂正
を実行
S24
訂正等を実施した後の
リード結果をストア
S25
終了

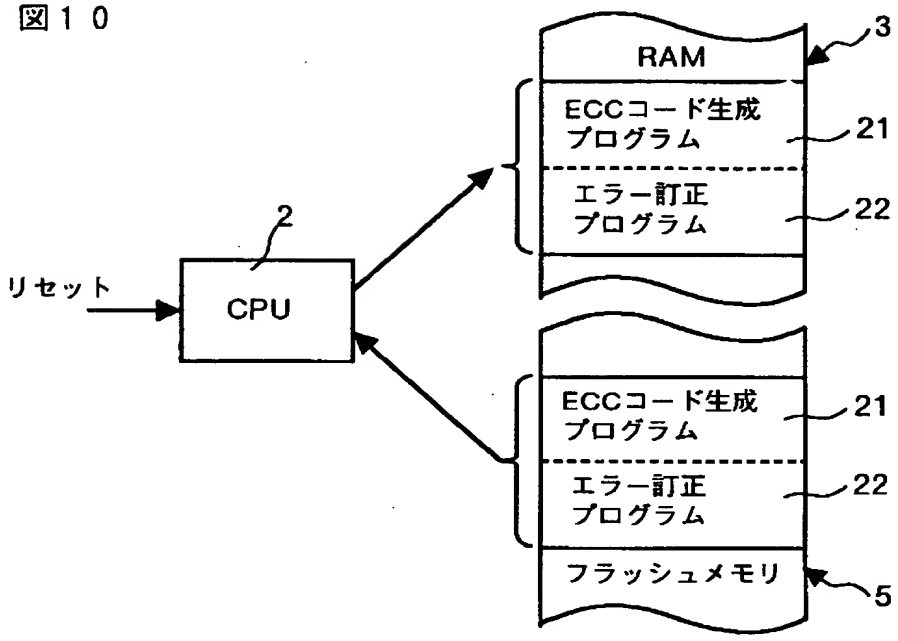
【図９】

図９


エラー訂正付きリードフロー
S31
ユーザデータ配列とECC
コード配列のリードアドレス
を別々に設定
リードアドレスに対応する
ＥＣＣデータとユーザデータ
を読み込む
S32
No
リードデータエラー訂正
を実施？
S33
Yes
リードデータのエラー訂正
を実行
S34
訂正等を実施した後の
リード結果をストア
S35
終了

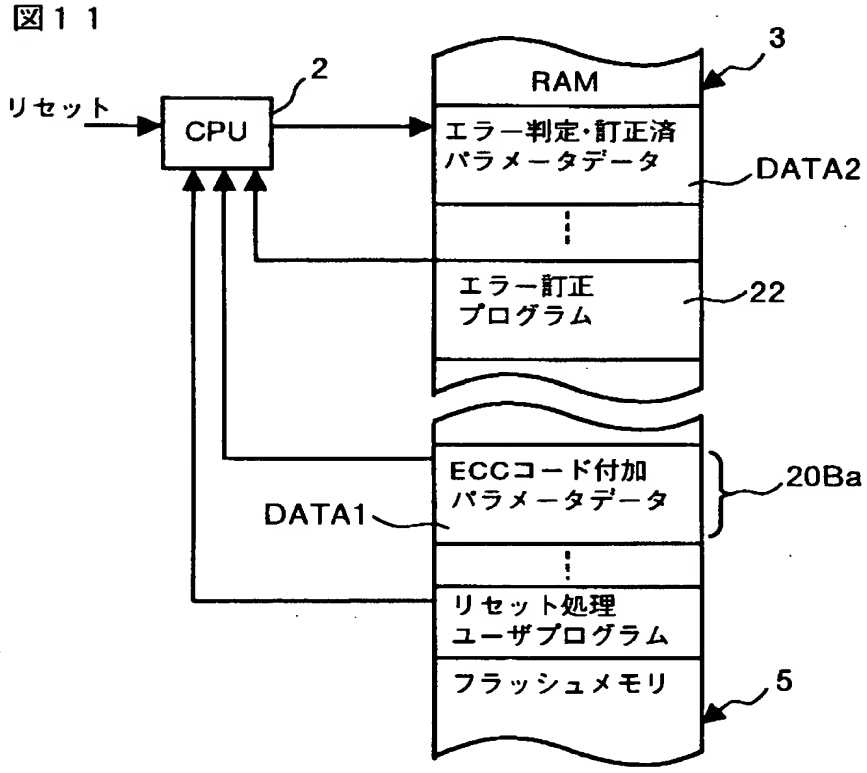
【図１０】

図１０



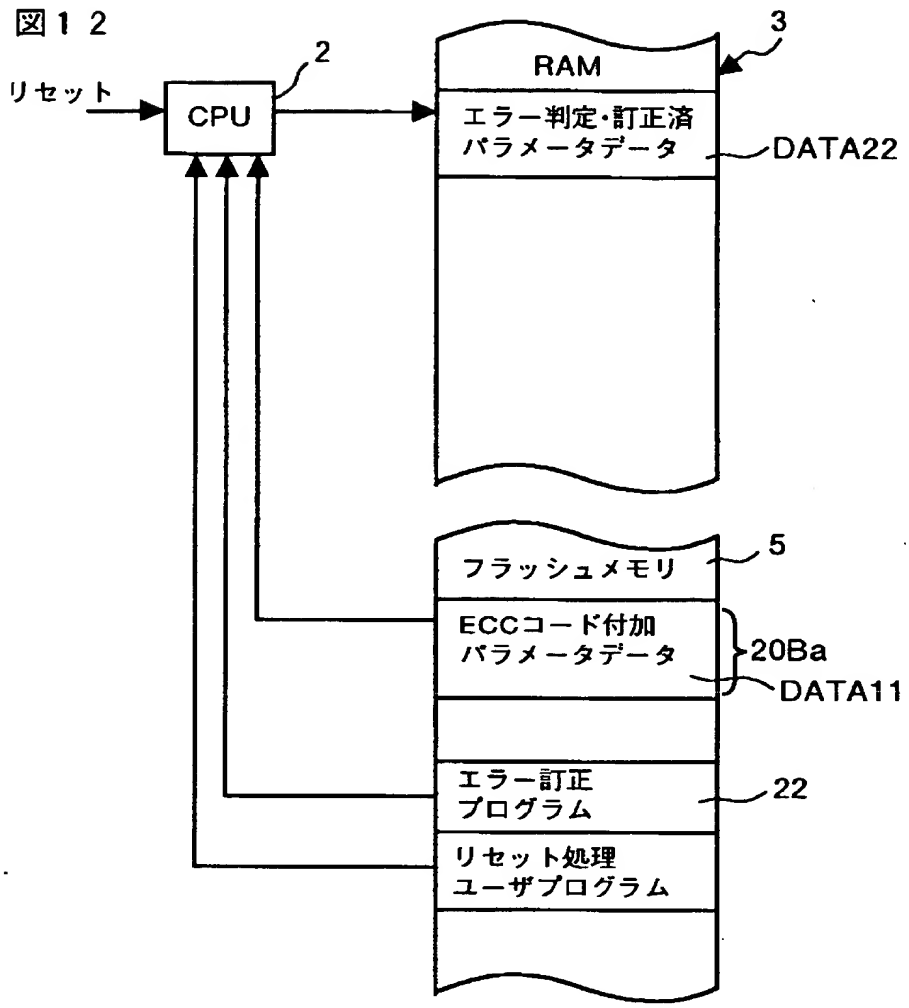
【図１１】

図１１



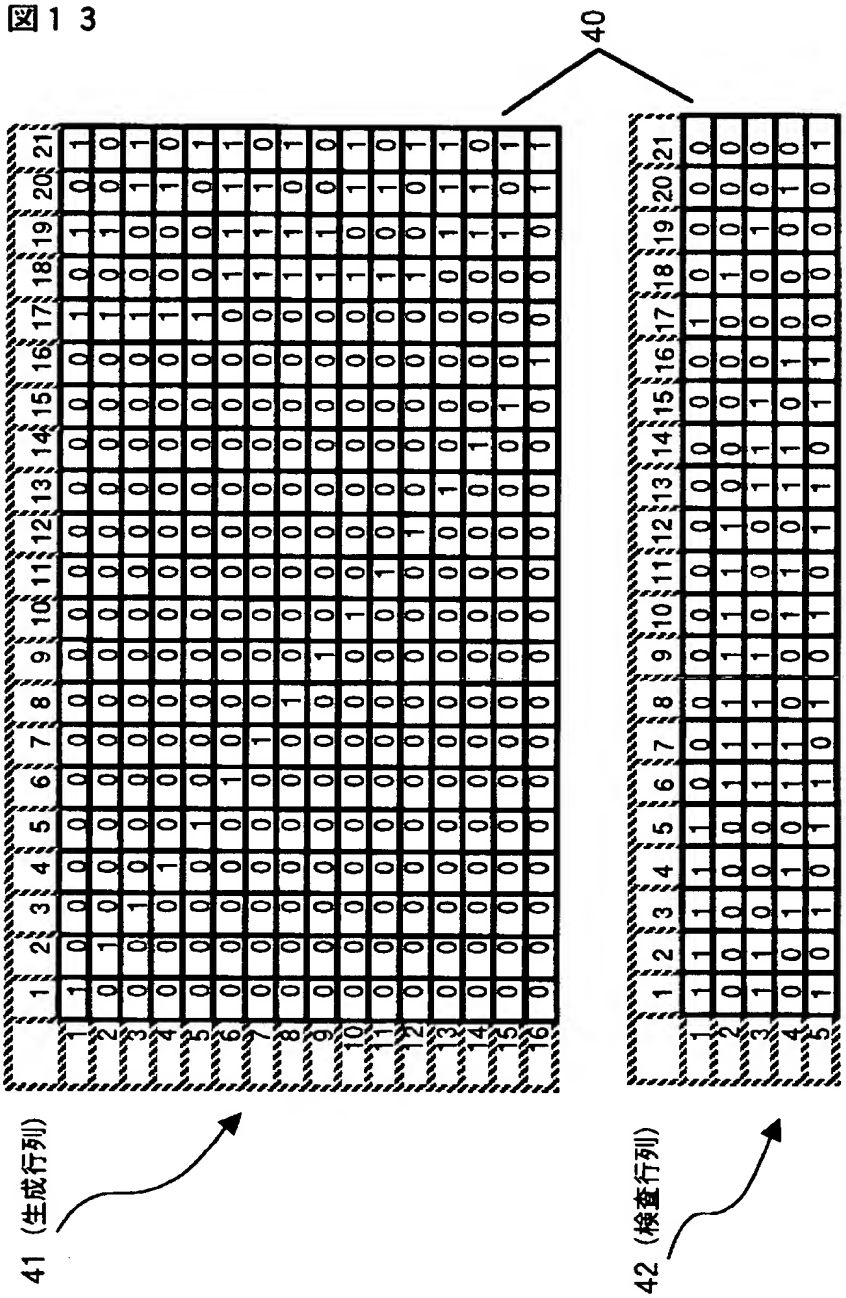
【図１２】

図１２


リセット
CPU
2
RAM
3
エラー判定・訂正済
パラメータデータ
DATA22
フラッシュメモリ
5
ECCコード付加
パラメータデータ
20Ba
DATA11
エラー訂正
プログラム
22
リセット処理
ユーザプログラム

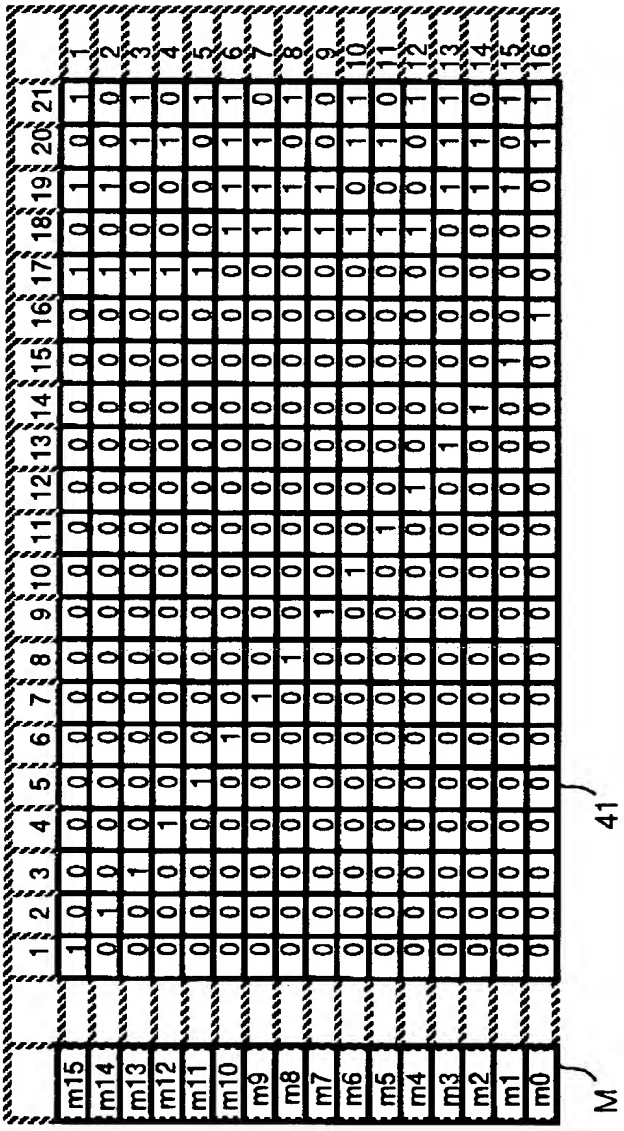
【図１３】

図１３



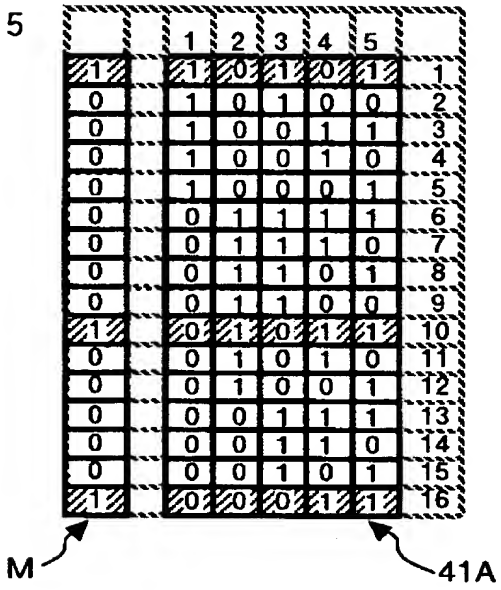
【図１４】

図１４



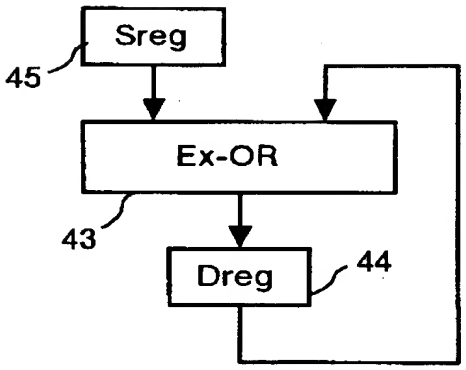
【図１５】

図１５



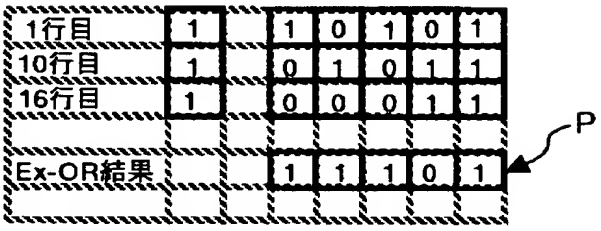
【図１６】

図１６



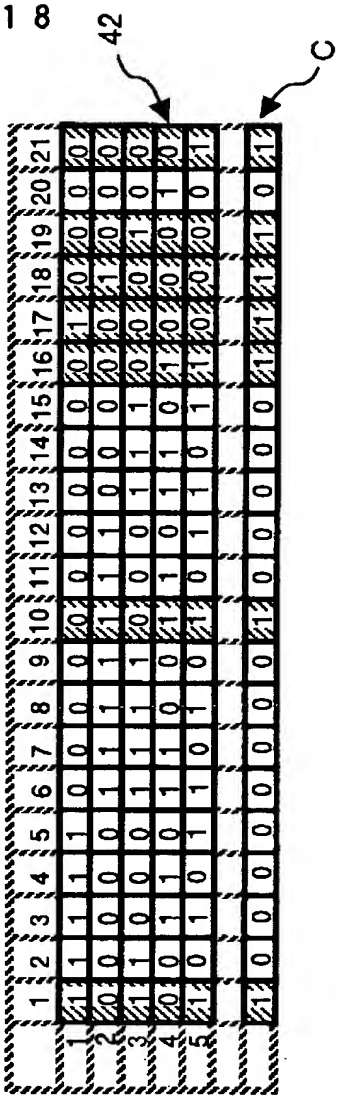
【図１７】

図１７



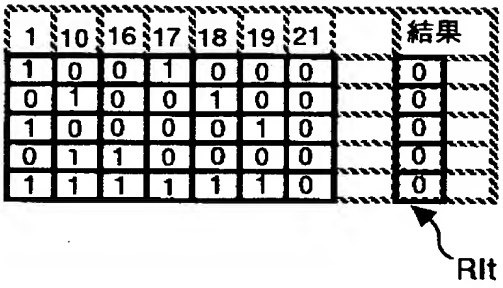
【図１８】

図１８



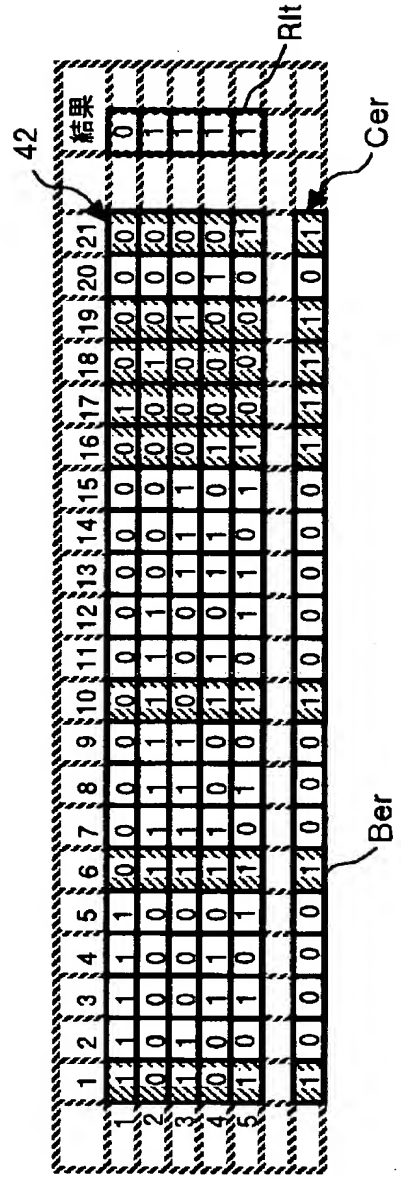
【図１９】

図１９



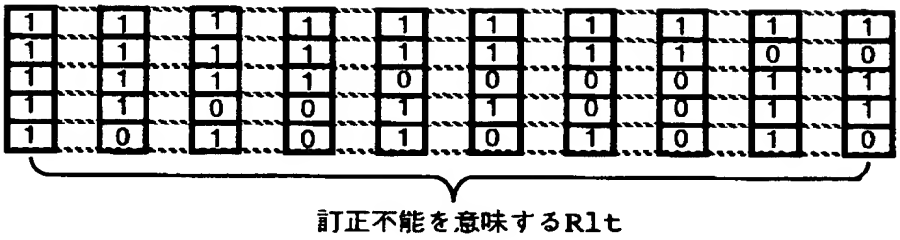
【図２０】

図２０



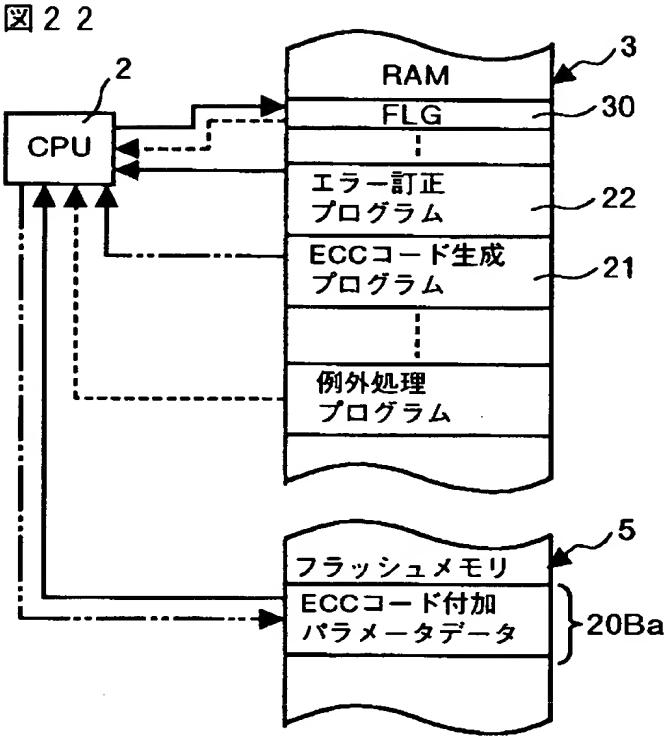
【図２１】

図２１



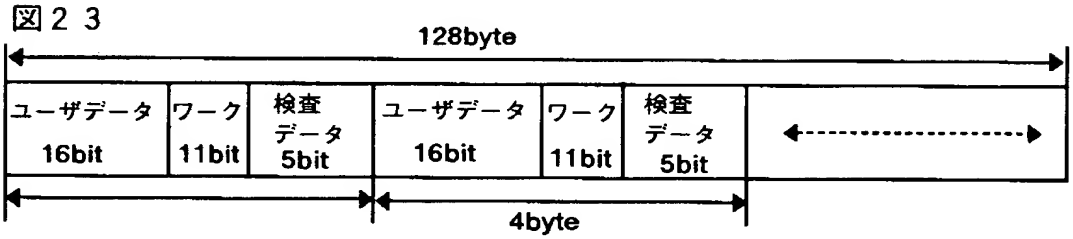
【図２２】

図２２



【図２３】

図２３

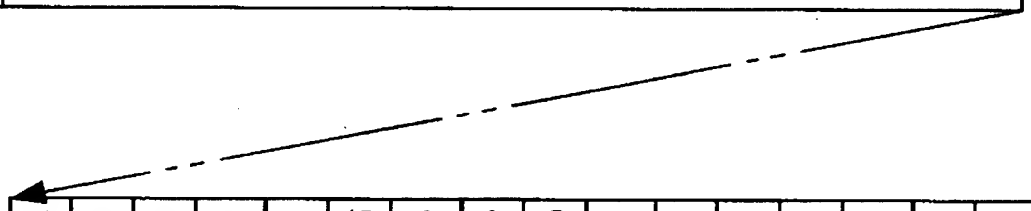


【図２４】

図２４

ECCデータブロックの構成

ビット	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
機能名	D15	D14	D13	D12	D11	D10	D09	D08	D07	D06	D05	D04	D03	D02	D01	D00
初期値	ユーザの書き込みデータ															
備考	ユーザの書き込みデータ（エラーコレクションを実施したいデータ）															



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	P04	P03	P02	P01	P00
ユーザは、ALL"1"を転送する。															
未使用											符号化検査ビット				

【図２５】

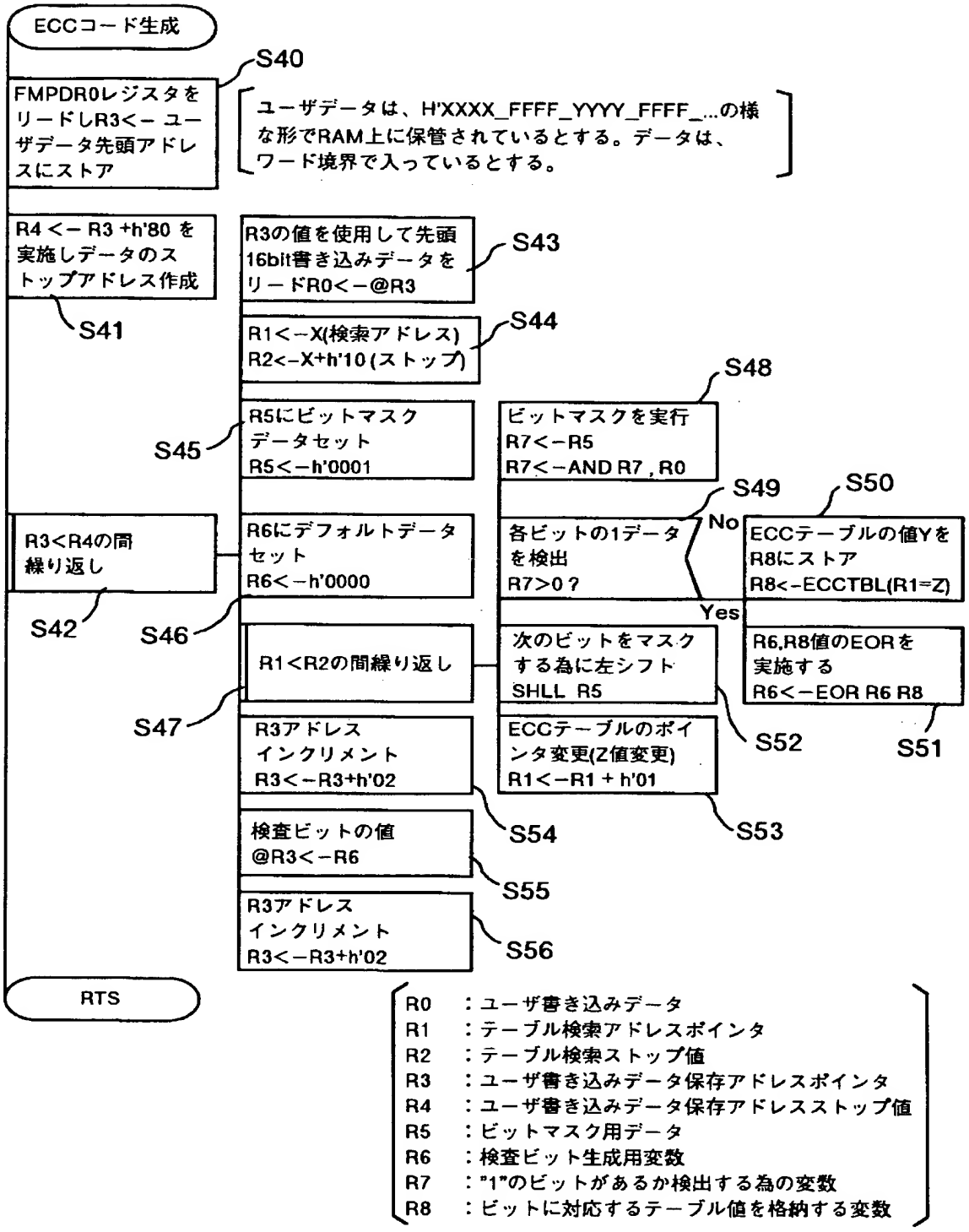
図２５

検索アドレス(Z)	生成用データ(Y)	機能名
X	b'0000_0011	D00
X+1	b'0000_0101	D01
X+2	b'0000_0110	D02
X+3	b'0000_0111	D03
X+4	b'0000_1001	D04
X+5	b'0000_1010	D05
X+6	b'0000_1011	D06
X+7	b'0000_1100	D07
X+8	b'0000_1101	D08
X+9	b'0000_1110	D09
X+A	b'0000_1111	D10
X+B	b'0001_0001	D11
X+C	b'0001_0010	D12
X+D	b'0001_0011	D13
X+E	b'0001_0100	D14
X+F	b'0001_0101	D15
X+10	b'0000_0001	P00
X+11	b'0000_0010	P01
X+12	b'0000_0100	P02
X+13	b'0000_1000	P03
X+14	b'0001_0000	P04

ECCテーブル(ECCTLB)

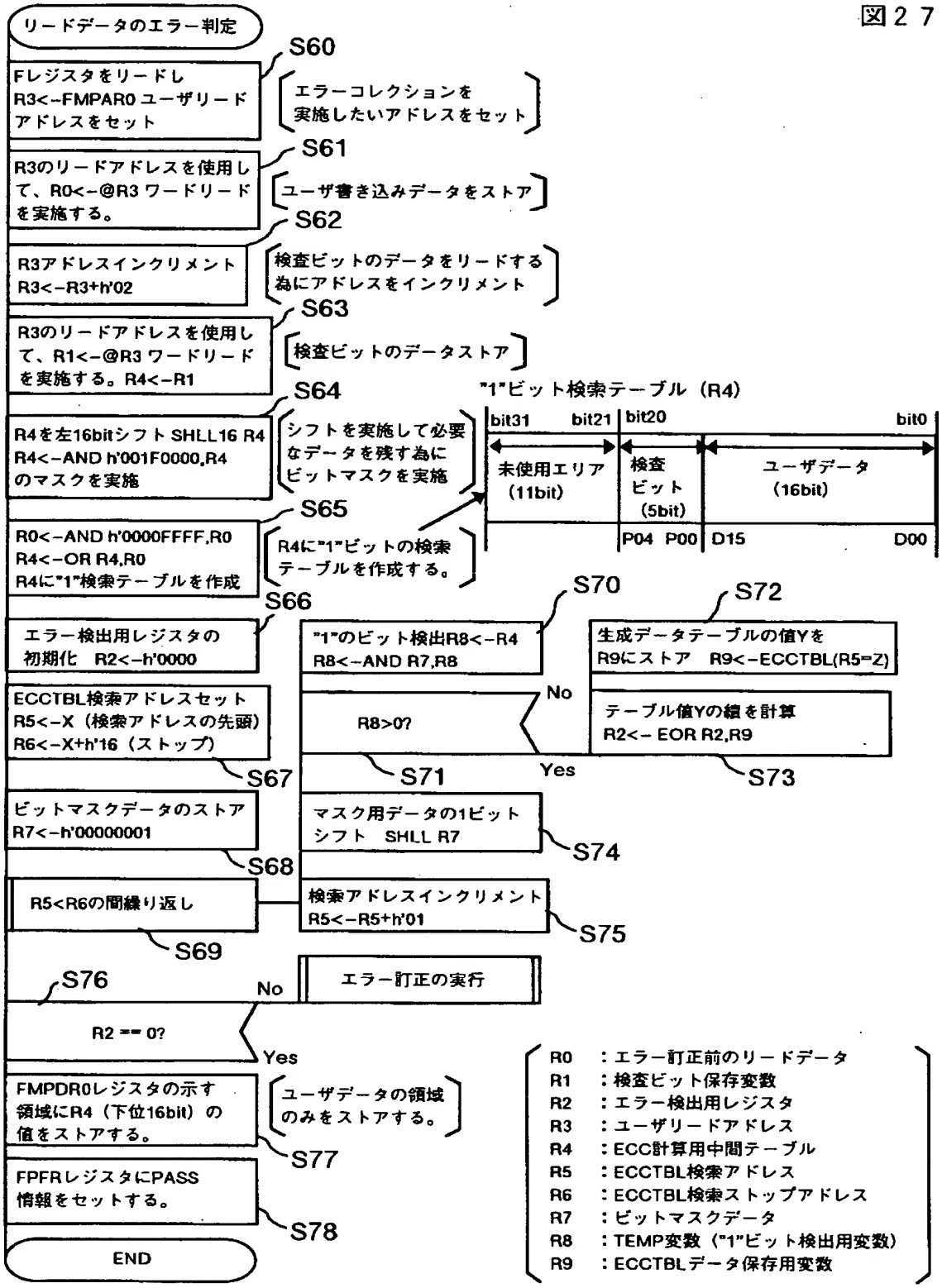
【図２６】

図２６


ECCコード生成
S40
FMPDR0レジスタをリードしR3<－ユーザデータ先頭アドレスにストア
ユーザデータは、H'XXXX_FFFF_YYYY_FFFF_...の様な形でRAM上に保管されているとする。データは、ワード境界で入っているとする。
R4 <－ R3 +h'80 を実施しデータのストップアドレス作成
S41
R3の値を使用して先頭16bit書き込みデータをリードR0<－@R3
S43
R1<－X(検索アドレス)
R2<-X+h'10 (ストップ)
S44
S48
S45
R5にビットマスクデータセット
R5<－h'0001
ビットマスクを実行
R7<－R5
R7<－AND R7 , R0
S49
S50
No
R3<R4の間繰り返し
R6にデフォルトデータセット
R6<－h'0000
各ビットの1データを検出
R7>0？
ECCテーブルの値Yを R8にストア
R8<-ECCTBL(R1=Z)
S42
S46
Yes
R1<R2の間繰り返し
次のビットをマスクする為に左シフト
SHLL R5
R6,R8値のEORを実施する
R6<－EOR R6 R8
S47
S52
S51
R3アドレスインクリメント
R3<－R3+h'02
ECCテーブルのポインタ変更(Z値変更)
R1<－R1 + h'01
S54
S53
検査ビットの値
@R3<－R6
S55
R3アドレスインクリメント
R3<－R3+h'02
S56
RTS
R0 ：ユーザ書き込みデータ
R1 ：テーブル検索アドレスポインタ
R2 ：テーブル検索ストップ値
R3 ：ユーザ書き込みデータ保存アドレスポインタ
R4 ：ユーザ書き込みデータ保存アドレスストップ値
R5 ：ビットマスク用データ
R6 ：検査ビット生成用変数
R7 ："1"のビットがあるか検出する為の変数
R8 ：ビットに対応するテーブル値を格納する変数

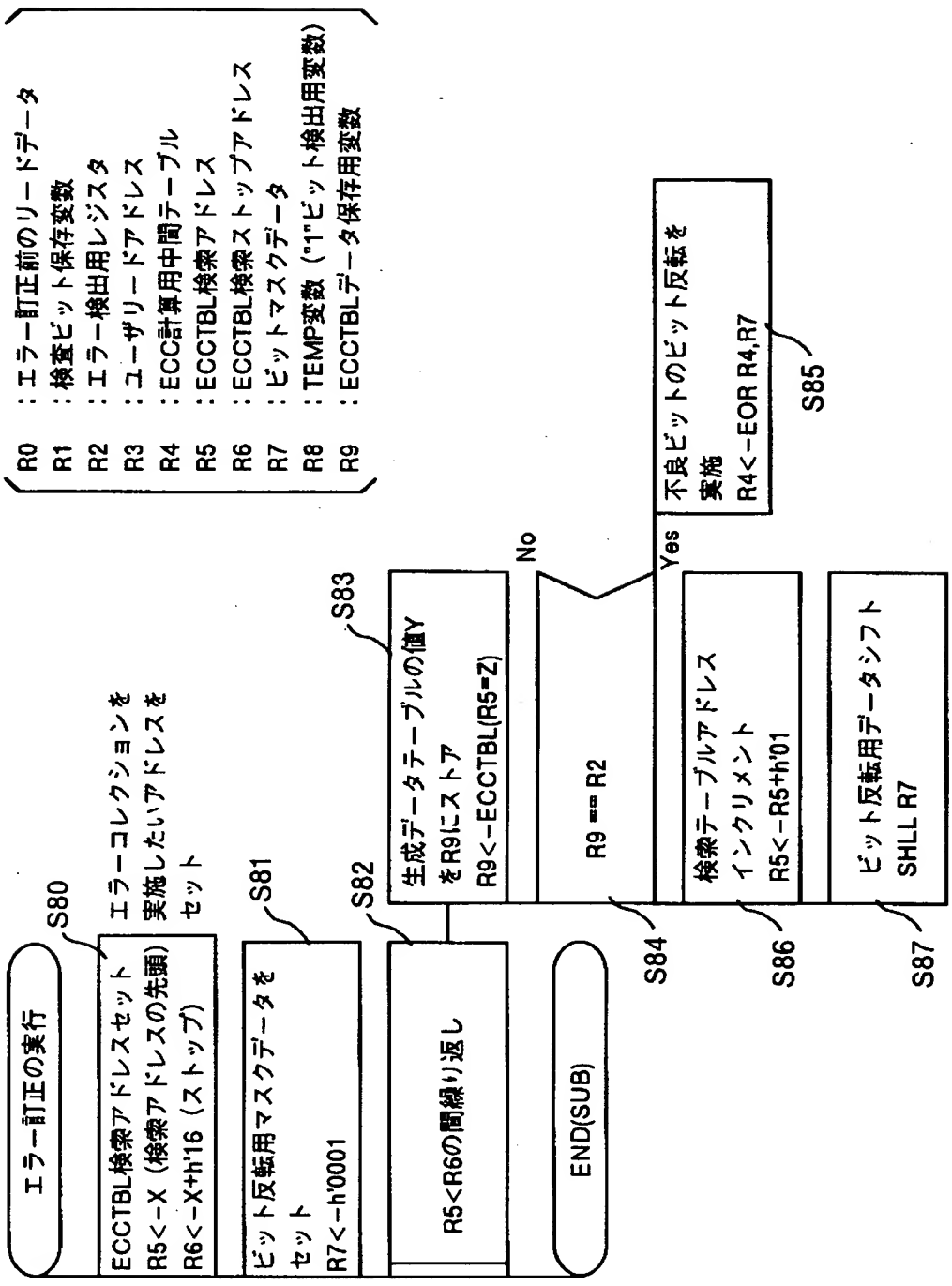
【図２７】

図２７


リードデータのエラー判定
S60
Fレジスタをリードし R3<-FMPAR0 ユーザリードアドレスをセット
エラーコレクションを実施したいアドレスをセット
S61
R3のリードアドレスを使用して、R0<-@R3 ワードリードを実施する。
ユーザ書き込みデータをストア
S62
R3アドレスインクリメント R3<-R3+h'02
検査ビットのデータをリードする為にアドレスをインクリメント
S63
R3のリードアドレスを使用して、R1<-@R3 ワードリードを実施する。R4<-R1
検査ビットのデータストア
S64
R4を左16bitシフト SHLL16 R4
R4<-AND h'001F0000,R4 のマスクを実施
シフトを実施して必要なデータを残す為にビットマスクを実施
"1"ビット検索テーブル（R4）
bit31
bit21
bit20
bit0
未使用エリア (11bit)
検査ビット (5bit)
ユーザデータ (16bit)
P04 P00
D15
D00
S65
R0<-AND h'0000FFFF,R0
R4<-OR R4,R0
R4に"1"検索テーブルを作成
R4に"1"ビットの検索テーブルを作成する。
S66
エラー検出用レジスタの初期化 R2<-h'0000
S70
"1"のビット検出R8<-R4
R8<-AND R7,R8
S72
生成データテーブルの値Yを R9にストア R9<-ECCTBL(R5=Z)
ECCTBL検索アドレスセット R5<-X（検索アドレスの先頭）
R6<-X+h'16（ストップ）
S67
R8>0?
No
Yes
S71
テーブル値Yの値を計算 R2<- EOR R2,R9
S73
ビットマスクデータのストア R7<-h'00000001
S68
マスク用データの1ビットシフト SHLL R7
S74
R5<R6の間繰り返し
検索アドレスインクリメント R5<-R5+h'01
S75
S69
S76
No
エラー訂正の実行
R2 == 0?
Yes
FMPDR0レジスタの示す領域にR4（下位16bit）の値をストアする。
ユーザデータの領域のみをストアする。
S77
FPFRレジスタにPASS情報をセットする。
S78
END
R0 ：エラー訂正前のリードデータ
R1 ：検査ビット保存変数
R2 ：エラー検出用レジスタ
R3 ：ユーザリードアドレス
R4 ：ECC計算用中間テーブル
R5 ：ECCTBL検索アドレス
R6 ：ECCTBL検索ストップアドレス
R7 ：ビットマスクデータ
R8 ：TEMP変数（"1"ビット検出用変数）
R9 ：ECCTBLデータ保存用変数

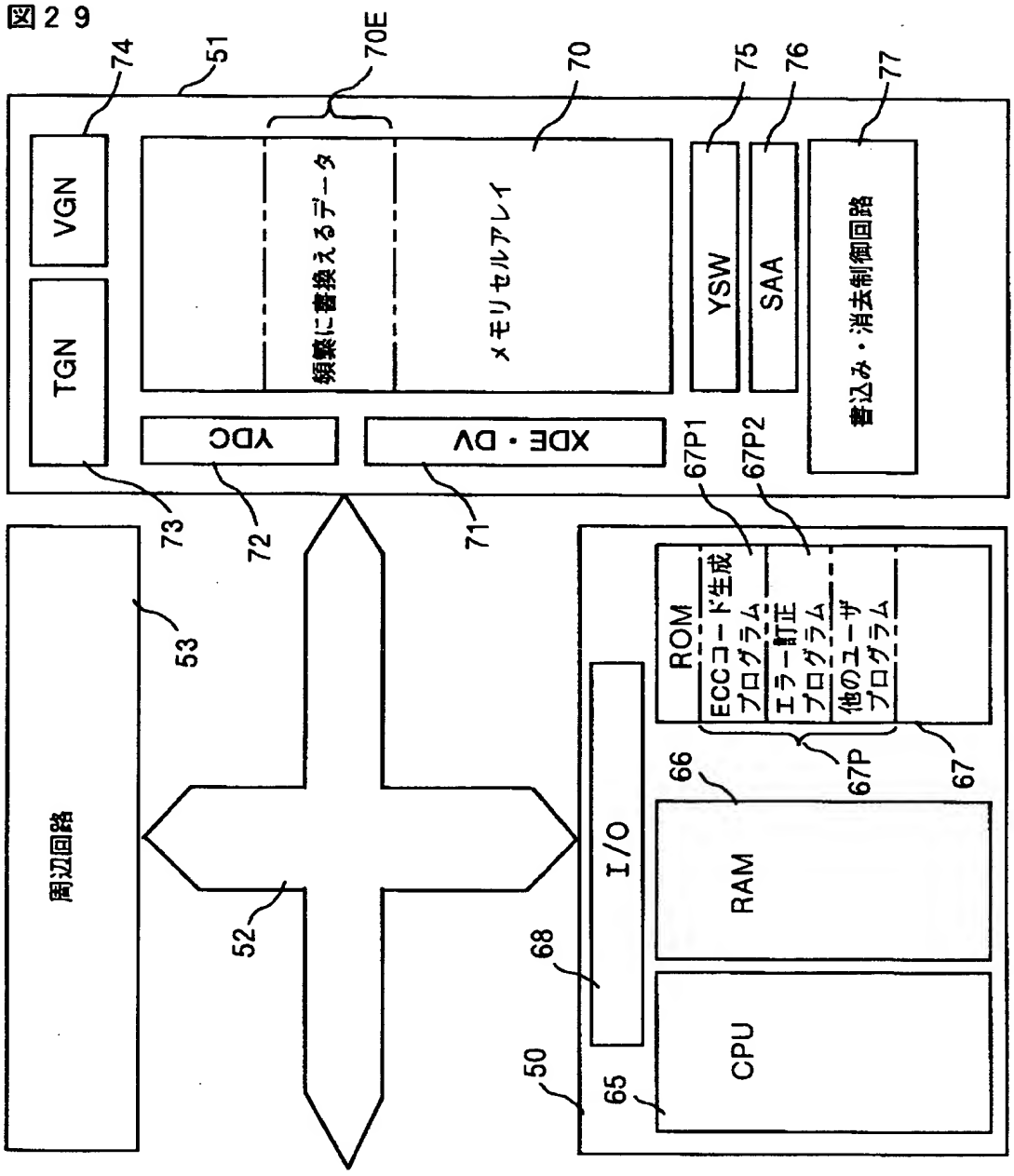
【図２８】

図２８


R0 ：エラー訂正前のリードデータ
R1 ：検査ビット保存変数
R2 ：エラー検出用レジスタ
R3 ：ユーザリードアドレス
R4 ：ECC計算用中間テーブル
R5 ：ECCTBL検索アドレス
R6 ：ECCTBL検索ストップアドレス
R7 ：ビットマスクデータ
R8 ：TEMP変数（"1"ビット検出用変数）
R9 ：ECCTBLデータ保存用変数
エラー訂正の実行
S80
ECCTBL検索アドレスセット
R5<−X（検索アドレスの先頭）
R6<−X+h'16（ストップ）
エラーコレクションを
実施したいアドレスを
セット
S81
ビット反転用マスクデータを
セット
R7<−h'0001
S82
R5<R6の間繰り返し
S83
生成データテーブルの値Y
をR9にストア
R9<−ECCTBL(R5=Z)
No
R9 == R2
Yes
S84
不良ビットのビット反転を
実施
R4<−EOR R4,R7
S85
検索テーブルアドレス
インクリメント
R5<−R5+h'01
S86
ビット反転用データシフト
SHLL R7
S87
END(SUB)

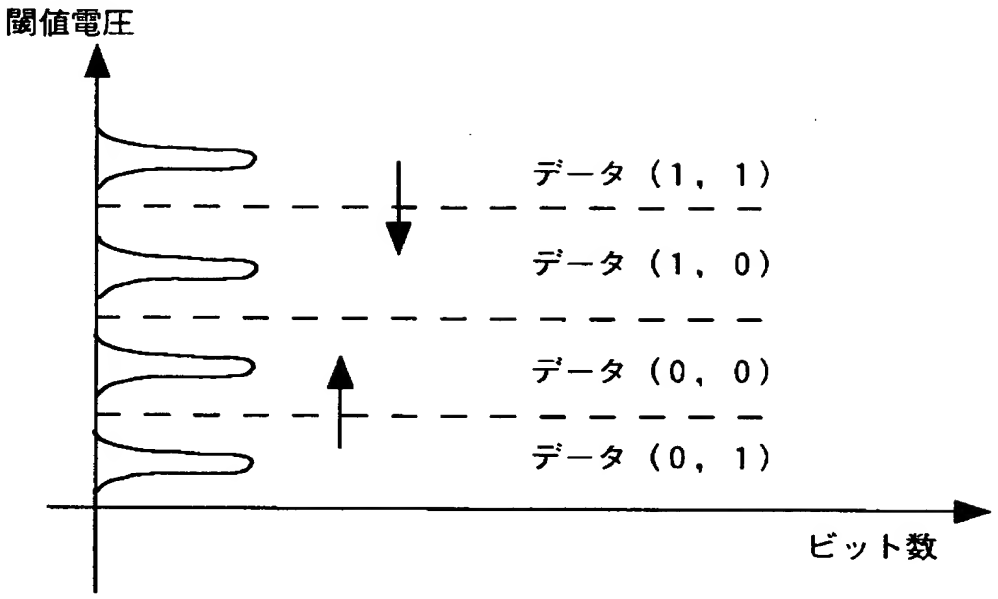
【図２９】

図２９


74
51
70E
70
75
76
77
VGN
TGN
頻繁に書換えるデータ
メモリセルアレイ
YSW
SAA
書込み・消去制御回路
YDC
XDE・DV
67P1
67P2
73
72
71
53
ROM
ECCコード生成プログラム
エラー訂正プログラム
他のユーザプログラム
67P
67
66
周辺回路
I/O
RAM
52
68
CPU
50
65

【図３０】

図３０



【書類名】　　要約書

【要約】

【課題】　ＥＣＣコードによる記憶領域の利用の無駄を省いて不揮発性メモリにおける記憶情報の書換え保証回数を向上させる。

【解決手段】　データ処理システム（１）は、書き換え可能な不揮発性メモリ（５）と、中央処理装置（２）とを有し、前記中央処理装置は、前記不揮発性メモリの指定された一部の記憶領域（２０Ｂａ）に対してだけソフトウェアＥＣＣ処理の対象とする。一部の記憶領域にだけＥＣＣコードの付加や誤り訂正を行なって書換え保証回数を向上させるから、記憶領域に拘わらず全てのライトデータに対して区別なくＥＣＣコードを付加する構成に比べて、実質的に無用のＥＣＣコードによる記憶領域の無駄を省くことができ、また、ＥＣＣ処理をソフトウェアで対処するから不揮発性メモリのデバイス特性に合ったＥＣＣ訂正能力を容易に選択することができる。

【選択図】　　図１

出　願　人　履　歴　情　報

識別番号　　〔０００００５１０８〕

１．変更年月日　　１９９０年　８月３１日
［変更理由］　新規登録
住　所　東京都千代田区神田駿河台４丁目６番地
氏　名　株式会社日立製作所

出　願　人　履　歴　情　報

識別番号　　〔０００２３３５９４〕

１．変更年月日　　１９９０年　８月３１日
［変更理由］　新規登録
住　所　　北海道亀田郡七飯町字中島１４５番地
氏　名　　日立北海セミコンダクタ株式会社